VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

INTEGRACE SYSTÉMU ROZŠÍŘENÉ REALITY DO TESTBEDU PRŮMYSL 4.0
AUGMENTED REALITY SYSTEM FOR INDUSTRY 4.0 TESTBED

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Matěj Poláček**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Václav Kaczmarczyk, Ph.D.**
SUPERVISOR

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Matěj Poláček ***ID:*** 192500
Ročník: 2 ***Akademický rok:*** 2019/20

NÁZEV TÉMATU:

Integrace systému rozšířené reality do testbedu Průmysl 4.0

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je navrhnout a realizovat mobilní aplikaci pro systém Android,

která vytvoří tzv. rozšířenou realitu nad testbedem Průmysl 4.0.

1. Seznamte se s koncepcí testbedu a požadavky na navrhované zařízení.
2. Proveďte zhodnocení a určete vhodné IDE pro návrh mobilních aplikací.
3. Proveďte průzkum a po konzultaci s vedoucím určete vhodnou knihovnu pro realizaci detekce objektů ve scéně.
4. Implementujte aplikaci ve zvoleném systému a otestujte ji.
5. Dokumentujte celé řešení už v průběhu celé práce.

DOPORUČENÁ LITERATURA:

[1] An Industry 4.0 Testbed (Self-Acting Barman): Principles and Design (Kaczmarczyk, 2018)

Termín zadání: 3.2.2020 ***Termín odevzdání:*** 1.6.2020

Vedoucí práce: Ing. Václav Kaczmarczyk, Ph.D.

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

ABSTRAKT

Diplomová práce se zabývá problematikou integrace rozšířené reality do testbedu Průmysl 4.0. Testbed představuje automatizovaného robotického barmana, který slouží k praktické demonstraci a ověření pojmů jako je Průmysl 4.0, virtuální zprovoznění či digitální továrna. Integrace rozšířené reality je provedena pomocí mobilní aplikace na operační systém Android a vytváří AR informační systém testbedu. Teoretická část práce je věnována Průmyslu 4.0, konstrukci testbedu, rozšířené realitě a technologiím použitých při implementaci aplikace. Druhá část se zabývá implementací a testováním aplikace. Na závěr je uvedeno zhodnocení výsledků práce.

KLÍČOVÁ SLOVA

Rozšířená realita, Android, Unity, ARCore, testbed Průmysl 4.0, REST API

ABSTRACT

The diploma thesis is concerned with the research of integration of augmented reality into the testbed Industry 4.0. Testbed is presenting automated robotic barman intended for practical demonstration and verification of concepts such as Industry 4.0 or a digital factory. The integration of augmented reality is realized via an Android application and creates the AR information system of the testbed. The theoretical part of the thesis is about Industry 4.0, the construction of the testbed and technologies of augmented reality used during implementation of the application. The practical part deals with the implementation and testing of the application. The conclusion includes an evaluation of the goals of the thesis.

KEYWORDS

Augmented reality, Android, Unity, ARCore, testbed Industry 4.0, REST API

BIBLIOGRAFICKÁ CITACE

POLÁČEK, Matěj. *Integrace systému rozšířené reality do testbedu Průmysl 4.0* [online]. Brno, 2020 [cit. 2020-05-03]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/127089. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Václav Kaczmarczyk.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „*Integrace systému rozšířené reality do testbedu Průmysl 4.0*“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom/a následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně 22. 5. 2020 ……

Podpis autora

PODĚKOVÁNÍ

Tímto chci poděkovat vedoucímu diplomové práce panu Ing. Václavu Kaczmarczykovi, Ph.D. za odborné vedení a pomoc během vývoje aplikace a psaní diplomové práce.

V Brně 22. 5. 2020 ……

Podpis autora

OBSAH

1 ÚVOD

V dnešní době jsme součástí čtvrté průmyslové revoluce, tedy Průmyslu 4.0. Propojením fyzického, kybernetického a virtuálního světa spolu se zavedením internetu, digitalizace a umělé inteligence do výroby se mění celá tvář průmyslu. Všechny tyto faktory výrazně zkracují čas potřebný k uvedení nového produktu na trh, zvyšují efektivitu výroby a celkovou kvalitu produktů [1].

Tato diplomová práce se zabývá návrhem a implementací mobilní aplikace na operační systém Android, která integruje rozšířenou realitu do projektu testbed Průmysl 4.0 s názvem Barman. Pojmem rozšířená realita je myšleno rozšíření reálného obrazu o doplňující digitální informace. Díky této technologii lze pro projekt Barmana vytvořit AR informační systém. Obsah tohoto systému je získáván z REST API testbedu, které bylo vytvořeno v rámci této diplomové práce.

Testbed je vyvíjen na Ústavu automatizace a měřicí techniky na Vysokém učení technickém v Brně. Jedná se o automatizovaného robotického barmana, který slouží k praktické demonstraci a ověření principů, jimiž jsou Průmysl 4.0, virtuální zprovoznění či digitální továrna. Vytvořená aplikace přidává testbedu novou funkcionalitu a rozšiřuje uživatelské rozhraní o další dimenzi. Doplňuje tak klíčové aspekty celého projektu o další prvek.

1.1 CÍL PRÁCE

Cílem této diplomové práce je vytvoření aplikace na operační systém Android, díky které bude do testbedu integrována rozšířená realita pomocí informačního systému.

S tímto úkolem je spojeno několik zásadních kroků:

- seznámení s koncepcí testbedu Průmysl 4.0 a technologií rozšířené reality;
- zhodnocení a výběr vhodného IDE pro vývoj mobilní AR aplikace na operační systém Android;
- zhodnocení a výběr vhodné platformy integrující rozšířenou realitu;
- návrh, implementace a testování aplikace;
- vytvoření dokumentace.

Základem mobilní aplikace bude možnost snímání scény, přesněji testbedu, pomocí videokamery zařízení a vykreslení AR prvků na displeji, které vytvoří AR informační systém. Úkolem systému je prezentování aktuálního stavu testbedu. Důraz musí být kladen na možnost konfigurace, řízení vzhledu a obsahu informačního systému ze strany serveru.

1.2 STRUKTURA PRÁCE

První část práce se zabývá teoretickým úvodem do řešené problematiky a výkladem pojmů a definic, které jsou následně aplikovány v části praktické. V začátku je věnován prostor tématu Průmysl 4.0 a představení projektu Barman. Následně je zde řešena problematika virtuality, která seznamuje se zásadními pojmy rozšířená, virtuální a smíšená realita.

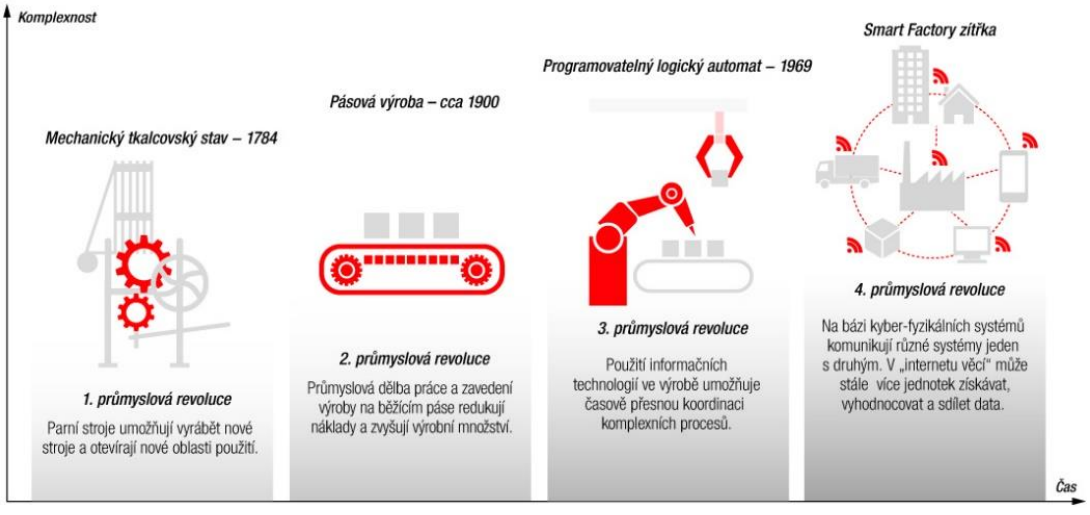
Posléze jsou uvedeny dvě analýzy, které již částečně spadají do praktické části. Náplní první je rozbor AR platforem použitelných k vývoji mobilní aplikace pro operační systém Android. Na ni navazuje detailní popis funkcí vybrané technologie ARCore. Druhá analýza se zabývá výběrem vhodného vývojového prostředí. Výsledkem je použití herního enginu Unity3D s integrovaným IDE.

Praktická část popisuje vývoj mobilní marker-based AR aplikace na operační systém Android integrující rozšířenou realitu do testbedu Průmysl 4.0. K tomuto účelu byl použit engine Unity ve spojení s SDK ARCore. Na úvod je představeno Unity vývojové prostředí s vysvětlením základních principů práce s ním. Je zde uveden postup k vytvoření základu projektu a konfigurace testovacího mobilního zařízení, které bylo během implementace použito. Dále je vysvětlen přístup k celé problematice integrace rozšířené reality. Je řešena problematika orientace mobilního zařízení v prostoru a s ním spojená registrace AR, určení pracovní plochy testbedu pomocí markeru a vytvoření AR informačního systému. Velký důraz byl kladen na styl zobrazování obsahu informačního systému. Cílem bylo vytvoření co nejvíce obecného řešení, aby v případě budoucích fyzických změn na stávajících komponentách testbedu není nutné vytvářet novou verzi aplikace. Obsah a důležitá nastavení informačního systému jsou získávány z REST API, které vzniklo v rámci této práce. Poslední část se zabývá testováním finální verze aplikace a zhodnocením výsledku.

2 PRŮMYSL 4.0

První průmyslová revoluce začala v druhé polovině 18. století v Anglii. Symbolem této revoluce se stal parní stroj, vynalezený v roce 1765. Tento vynález a další nové objevy byly postupně nasazeny v průmyslové výrobě a přinesly velké zrychlení a zvýšení produktivity práce. Dopad průmyslové revoluce na společnost byl významný a zásadně se změnily všechny obory hospodářství. Druhá průmyslová revoluce se datuje na konec 19. století a bezprostředně tak navazuje na období po první průmyslové revoluci. Počátečním impulzem byly elektrifikace, vznik montážních linek a dělba práce. Mezi hlavní průkopníky patřil Henry Ford, který využitím montážní linky a pásové výroby výrazně změnil proces výroby aut. Během třetí průmyslové revoluce pak byla ve větší míře nahrazována lidská práce automatizací procesů a docházelo k integraci informačních technologií a robotiky do výroby. [2]

V současné době jsme svědky čtvrté průmyslové revoluce. Moderní technologie ovlivňují nejen každodenní způsob života, ale i celou podobu průmyslu, viz Obr. 2-1. Řada států již na tuto situaci reagovala a přišla s vlastní iniciativou, jak tento pokrok využít ve svůj prospěch. Takto zapojené státy nyní vytváří vlastní programy s jasným cílem, kterým je posílení jejich konkurenceschopnosti a technologického prvenství na světovém trhu. Jednotlivé státy vytváří velké dotační a investiční plány na podporu a rozvoj technologií. Právě Průmysl 4.0 je evropskou iniciativou. Poprvé byl prezentován na veletrhu v německém městě Hannover v roce 2011 a o dva roky později zde byl také zahájen. [1]



Obr. 2-1 Vizualizace etap vývoje průmyslové výroby [3]

2.1 CÍL PRŮMYSLU 4.0

Hlavním cílem Průmyslu 4.0 je vytvořit optimalizované výrobní prostředí, které je převedeno ze samostatných a izolovaných elementů na prostředí s plně automatizovanými jednotkami. Tyto systémy umožňují strojové vnímání, podporu dělníka a jsou mezi sebou vzájemně propojeny za účelem automatické optimalizace, konfigurace a diagnostiky. Vzájemným propojením jednotek prostřednictvím datové infrastruktury vznikají nové globalizované vysokorychlostní sítě, které tvoří kyberneticko-fyzické systémy CPS (*Cyber-Physical Systems*). CPS pak vytvářejí základ k vytvoření „inteligentních továren“, které jsou schopny autonomní výměny informací a mohou spolu kooperovat na základě svých potřeb. [1]

Díky informacím, které jsou o vyráběných produktech shromažďovány a ukládány, vznikají „inteligentní výrobky“. Takový výrobek přesně zná svou výrobní historii, je jednoznačně identifikovatelný a lokalizovatelný. Výrobní linky mohou v reálném čase reagovat na individuální požadavky aktuální poptávky i na případné chyby nebo poruchy kooperujících výrobních linek. [1]

2.2 TESTBED PRŮMYSL 4.0

Cílem této kapitoly je seznámit čtenáře s automatizovaným robotem testbed Průmysl 4.0 nazývaným Barman. K pochopení řešené problematiky je zde rozebrána jeho základní konstrukce a popis jednotlivých částí, na které je následně aplikována praktická část diplomové práce.

2.2.1 Konstrukce

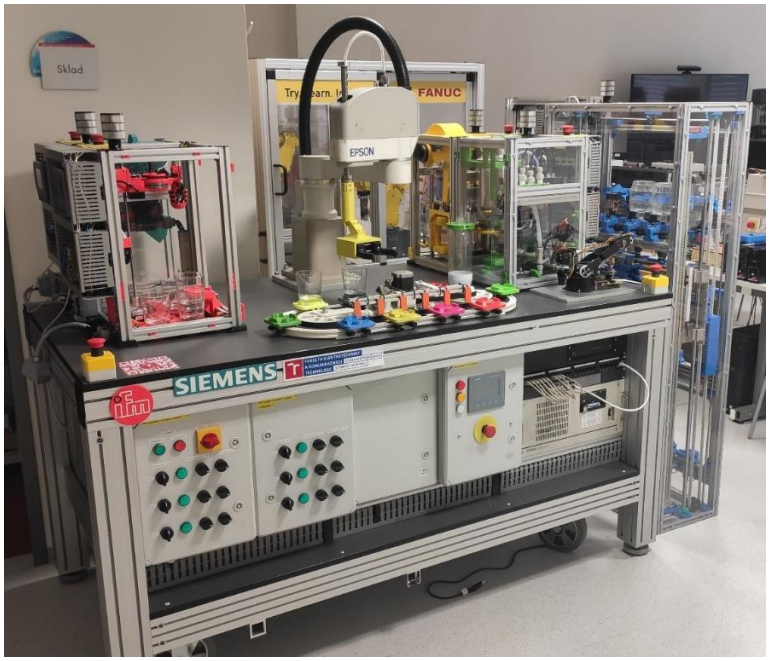
Východiskem diplomové práce je testbed Průmysl 4.0 - Barman. Jedná se o automatizovaného robotického barmana, který slouží k praktické demonstraci a ověření principů Průmysl 4.0, virtuální zprovoznění či digitální továrna. [4]

Barman je vyvíjen na Ústavu automatizace a měřicí techniky na Vysokém učení technickém v Brně. Je rozdělen do několika autonomních částí (buněk), které jsou obsluhovány robotickým ramenem a dopravníkem. Jednotlivé buňky jsou s veškerým svým podpůrným vybavením zakomponovány do konstrukce pracovního stolu o rozměrech 2000 x 1000 x 900 mm. Každá tato jednotka zajišťuje specifickou část procesu výroby míchaného nápoje.

Jedná se o tato zařízení:

- výrobník a zásobník perlivé vody s chlazením a automatickým dávkovačem sody do připravené nádoby;
- sklad s přesným dávkovačem ledu, kde je dávkované množství stanoveno kontinuálním vážením plněné nádoby;
- zásobník čistých a použitých sklenic s vlastním robotickým manipulátorem, přičemž zespod každé sklenice je umístěn NFC čip, na kterém je uložena požadovaná receptura k přípravě nápoje;
- sklad s láhvemi obsahující alkohol a sirupy s vlastním robotickým rovinným manipulátorem a dávkovačem požadovaného množství tekutiny;
- automatický homogenizátor obsahu sklenic;
- robotický manipulátor, který slouží k přesunu míchaných nápojů a zajišťuje tak distribuci jak hotových nápojů, tak přesun nádob mezi jednotlivými buňkami při procesu výroby;
- dopravní pás pro agregaci a vydávání hotových nápojů.

Čtyři autonomní buňky, každá o velikosti 330 x 330 x 500 mm, jsou umístěny na pracovní desce a mají shodný tvar. Konkrétně se jedná o výrobník sody, drtič ledu, sklad sklenic a míchač nápojů. Buňka se skladem lahví je kvůli svým větším rozměrům 1600 x 760 x 330 mm umístěna vedle konstrukce stolu, [4] viz. Obr. 2-2



Obr. 2-2 Pohled z předu na Testbed Průmysl 4.0

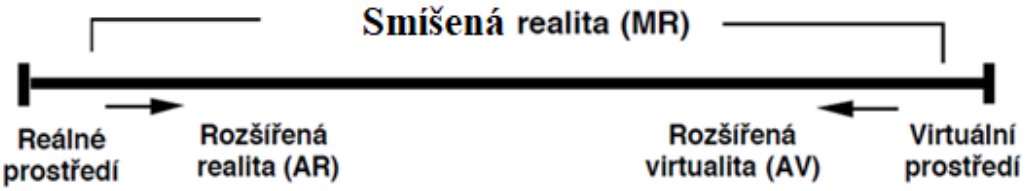
2.2.2 Proces výroby nápoje

Celý proces výroby začíná ve webové aplikaci, kde si uživatel nakonfiguruje požadovaný nápoj a následně požadavek odešle ke zpracování do testbedu. Výroba nápoje vychází z dat, která jsou uložena v NFC čipu zespod každé sklenice, který obsahuje jak recepturu nápoje, tak potřebné kroky výroby. Autonomní buňky čtou požadovaný postup výroby z tohoto čipu. Sklenice, ve které postupně vzniká nápoj, je přesouvána do jednotlivých buněk pomocí robotického ramene typu SCARA. Po finalizaci procesu je nápoj přemístěn do distribuční části robota. [4]

3 VIRTUALITA

V počátcích vývoje byly virtuální realita (virtual reality, VR), rozšířená realita (augmented reality, AR) a smíšená realita (mixed reality, MR) vnímány spíše jako nový doplněk zábavního průmyslu. Postupem času si však tyto technologie našly své uplatnění i v oblastech jako je vzdělávání, zdravotnictví, stavitelství, letectví nebo průmysl. Všechny typy zmíněných realit lze chápat jako počítačová rozhraní, která představují jeden z možných způsobů interakce uživatele s digitálním zařízením. Propracovanější imerzní systémy kooperují i s haptickou senzorikou, kdy se uživateli dostává naprosto věrohodnému pocitu, že je virtuální prostředí skutečné.

K pochopení propojení virtuálního a reálného prostředí v roce 1974 vytvořili Paul Milgram a Fumio Kishino koncept nazývaný reálně-virtuální kontinuum, viz. Obr. 3-1. Diagram zobrazuje hranice a vzájemný vztah mezi reálným a virtuálním prostředím. Na jedné straně kontinua je zobrazeno reálné prostředí složené pouze z reálných objektů a naopak, na straně druhé, prostředí virtuální, složené pouze z uměle vytvořených objektů. Mezi těmito dvěma světy se nachází rozšířená realita a rozšířená virtualita. Rozdíl mezi nimi je pouze základní prostor, který využívají jako primární prostředí. Pro rozšířenou realitu se jedná o reálné prostředí a u rozšířené virtuality o prostředí virtuální. K těmto prostředím se následně přidává množina virtuálních, resp. reálných objektů. Pro zjednodušení se rozšířená realita a rozšířená virtualita často spojuje pod jeden termín smíšená realita. [5]



Obr. 3-1 Grafické schéma reálně-virtuálního kontinua [5]

3.1 VIRTUÁLNÍ REALITA

Virtuální realita je technologie, která ztvárňuje uměle vytvořenou trojrozměrnou interaktivní skutečnost pomocí počítačové simulace v reálném čase. Tato imerzní technologie je kombinována se zobrazovacím zařízením, které uživatele zcela „vytrhne“ z reálného světa a přenese ho do světa syntetického. Pozorovateli se díky vizuálním, sluchovým a hmatovým efektům dostává pocitu, že je tímto prostorem zcela obklopen. [5]

3.1.1 Historie

Virtuální realita rozhodně není tématem posledních několika let, první náměty s moderním chápáním VR se totiž objevily již v 50. letech 20. století. V roce 1962 průkopník VR a filmař Morton Leonard Heilig zkonstruoval prototyp virtuální reality zvané Sensorama. Toto zařízení při promítání umožňovalo vnímat trojrozměrný obraz, prostorový stereo zvuk, vibrace a vůni. [6]

Postupem času vznikaly pokročilejší zobrazovací systémy. V roce 1980 Eric Howlett představil optický sytém LEEP (Large Expanse Extra Perspective), který se stal předlohou pro většinu dnes známých VR head-setů [7]. Největší vzestup moderních imerzních VR head-setů přinesl projekt Oculus Rift, který se dokázal prosadit díky v crowdfundingové sbírce na portálu Kickstarter, kde získal několikanásobně větší finanční obnos, než byl původní plán. K rozmachu VR přispěl vývoj výkonného a kompaktního hardware, který byl schopen nabídnout dostatečný výpočetní výkon k vykreslení potřebných detailů, čímž se nošení head-setu stalo více uživatelsky přívětivé, viz. Obr. 3-2.

Obr. 3-2 Oculus Quest a snímače pohybu [8]

3.2 ROZŠÍŘENÁ REALITA

Rozšířenou realitu (Augmented Reality, AR) je možné definovat jako krok mezi reálným světem a virtuální realitou, jak znázorňuje reálně-virtuální kontinuum, viz. Obr. 3-1.

Klíčové předpoklady, které definují rozšířenou realitu jsou: [5]

- interaktivita v reálném čase;
- registrace v trojrozměrném prostoru;
- spojení reálných a virtuálních objektů.

Principem rozšířené reality je schopnost doplnit realitu pozorovanou uživatelem virtuálními objekty. V nejlepším případě by se pozorovateli rozšířené reality mělo dostávat dojmu, že spolu reálné a virtuální objekty koexistují. Tohoto vjemu je typicky docíleno pomocí AR aplikace, která vykresluje syntetické objekty a umisťuje je do trojrozměrné scény reálného světa. Aby vjem rozšířené reality byl co nejvíce věrohodný, je důležité se zaměřit především na detaily vykresleného objektu, stíny a umístění, což s sebou nese velké nároky na výpočetní výkon zobrazovacího zařízení. [5]

3.2.1 Historie

První prototyp rozšířené reality pochází z roku 1968, kdy počítačový grafik Ivan Sutherland během studia na Harvardské univerzitě vytvořil první náhlavní displej (head mouted displey, HMD) spolu s podpůrným systémem zvaným Damoklův meč. Kvůli váze a velkým rozměrům musel být celý mechanismus upevněn ke stropu. Pomocí dvou obrazovek na brýlích byl vytvářen stereoskopický obraz zobrazující jednoduché geometrické obrazce na reálném okolí. [8]

3.2.2 Zobrazení rozšířené reality

Existují dva principy prezentace rozšířené reality a s ním spojená konfigurace AR aplikace: [8]

- optické zobrazení (optical-see through);
- video zobrazení (video-see through).

Optické zobrazení

U tohoto způsobu přímého pohledu uživatele se k zobrazení AR používá průhledových displejů. Uživateli se promítají virtuální objekty rozšířené reality na náhlavní displej

(HMD). Uživatel pozoruje skutečné prostředí skrz průhledové brýle. Spojením obrazu brýlí a obrazu skutečného světa uživatel získává vjem rozšířené reality [8], viz. Obr. 3-3.

Video zobrazení

U tohoto typu nepřímé vizualizace rozšířené reality jsou generované vizuální objekty vloženy do video obrazu pořízeného kamerou zobrazovacího zařízení, který je následně zobrazován na displeji. Pro tento účel lze velmi efektivně využívat tablet nebo chytrý telefon, viz. Obr. 3-4. Uváděný přístup je využíván u implementované aplikace v praktické části této diplomové práce. [8]

Obr. 3-3 Optické průhledové zobrazení [9]

Obr. 3-4 Marker a video průhledové zobrazení [10]

3.2.3 Pozicování a polohování v prostoru

Důležitou součástí AR technologie je lokalizace zobrazovacího zařízení v reálném prostředí. Existují dva základní přístupy, jak s touto problematikou pracovat [8].

Jedná se o:

- marker-based AR;
- markerless AR.

Maker-based AR

Součástí AR aplikace je sada definovaných značek – markerů, které slouží jako orientační body v prostoru. Marker s jednoznačným ohraničením je tvořen kontrastním piktogramem složeným z jednoduchých geometrických tvarů. Tento obrazec je možné snadno zachytit a rozpoznat ve snímané scéně. Po zachycení AR aplikací je na místo markeru umístěn virtuální objekt, viz Obr. 3-4. [8]

Markerless AR

U tohoto přístupu se zobrazovací zařízení v prostoru orientuje bez přídavných markerů. Technologie se spoléhá především na senzoriku zobrazovacího zařízení a na sofistikovaný software s důrazem na počítačové vidění. AR aplikace si kombinací dat z kamery a z IMU zařízení utváří vlastní trojrozměrnou virtuální představu o snímaném prostoru, ve kterém dokáže například detekovat horizontální, vertikální nebo nakloněné roviny. S touto znalostí ploch je umožněno umisťovat virtuální objekty do scény s větší věrohodností. [8]

3.2.4 Využití rozšířené reality v Průmyslu 4.0

Širokou možnost uplatnění rozšířené reality nabízí průmysl. Použití trojrozměrných modelovacích technik, simulací a vizualizací lze uplatnit například do procesu přípravy nového produktu. Může se jednat o části procesu jako je produktový design, procesní modelování, trénink nových pracovníků, testování nebo ověřování výsledků, viz Obr. 3-3.

Příklady využití rozšířené reality:

- konstruktérům a designerům je umožněno prohlédnout si finální produkt, aniž by musel být vyroben prototyp;
- v rámci týmu lze snáze pochopit a odhalit případné nedostatky;
- finální produkt může být mnohem lépe testován, neboť jakoukoliv změnu je možné okamžitě promítnout;
- prodejcům se nabízí velmi silný nástroj ke zlepšení prezentace výrobků;
- lze lépe experimentovat a není nutné vytvářet drahé fyzické prototypy produktů;
- vizualizace dat strojů při kontrole v pracovním prostředí;
- nástroj k zjednodušení servisu strojů;
- efektivnější zaškolování nového personálu s prezentováním postupu práce.

Tímto přístupem se dají ušetřit náklady a zkrátit čas potřebný k uvedení nového produktu na trh. To vše lze promítnout do snížení výsledné ceny výrobku a tím zvýšení zisku firmy a vrácení počátečních investic do nových technologií. Všechny tyto faktory jsou klíčové pro chod firmy a k získání náskoku před konkurencí. [11]

3.3 SMÍŠENÁ REALITA

Technologie smíšené reality (mixed reality, MR) využívá virtuální i rozšířené reality, jak představuje vizualizace reálně-virtuálního kontinua na Obr. 3-1. Tento termín zahrnuje rozšířenou realitu i rozšířenou virtualitu. Ke sloučení dochází v reálném čase a v reálném prostředí. Ve smíšené realitě je pozorovateli umožněno interakce a manipulace s virtuálními i fyzickými objekty. Je zde kladen důraz na sofistikovanou práci s reálným prostředím s využitím pokročilé senzoriky a optiky zobrazovacího zařízení.[8] Nejznámějším zastupitelem této kategorie je technologie HoloLens od společnosti Microsoft.

4 ANALÝZA VÝVOJOVÝCH AR PLATFOREM

K integraci rozšířené reality se využívají frameworky, které se specializují na AR aplikace. Mezi nejrozšířenější SDK pro operační systém Android jsou Google ARCore, ARToolkit nebo Vuforia od společnosti PTC.

4.1 ANDROID OS

Operační systém Android byl představen v roce 2003 stejnojmennou společností Android. Jejími zakladateli byli Andy Rubin, Rich Miner, Nick Sears a Chris White. V roce 2005 byla společnost Android Inc. koupena firmou Google. V roce 2007 bylo vytvořeno konsorcium Open Handset Alliance (OHA), které stojí v čele se společností Google za vývojem operačního systému. Konsorcium OHA je uskupení výrobců mobilních zařízení, telekomunikačních operátorů a technologických firem. První smartphone s OS Android se začal prodávat v roce 2008. Doposud je OS Android nejrozšířenějším operačním systémem pro chytrá zařízení. Tento operační systém najdeme nejen na mobilních zařízeních, ale také v tabletech, televizích, hodinkách nebo v automobilech. Nejnovější verzí je Android 10.

Operační systém tvoří vrstvu mezi hardwarem a uživatelem. Operační systém Android je postaven na linuxovém jádře a je dostupný jako open-source. Aplikace na operačním systému nekomunikují přímo s jádrem OS, ale pomocí jednotného API, přes kterého je možné přistupovat ke všem funkcím telefonu.

Obr. 4-1 OS Android 10 logo

4.2 ARTOOLKIT

ARToolKit je open-source multiplatformní SDK určené k vývoji aplikací s rozšířenou realitou. Knihovnu vyvinul Hirokazu Kato v roce 1999. Jedná se o velmi robustní a nejdéle vyvíjenou AR technologii. Podporuje multiplatformní vývoj marker-based aplikací. Nabízen je pod licencí GNU, avšak rozsah jeho funkcí je velmi omezen. Na trhu jsou nabízeny sofistikovanější SDK, než je tento.

4.3 VUFORIA

Vuforia obsahuje komplexní sadu nástrojů pro vývoj aplikací s rozšířenou realitou pro mobilní telefony, tablety či VR head-set s podporou široké palety vývojových prostředí a multipalformního vývoje. Vývojářům se otevírá možnost přidání principů počítačového vidění k rozpoznání a sledování obrázků (Image Targets), 3D objektů (Model Targets), markerů (VuMark) a textur (Cylindrical mapping).

V původním návrhu aplikace v teoretické části této práce byl použit framework Vuforia, avšak během práce na projektu došlo ke změně jeho licenčních podmínek a zpoplatnění. Na základě konzultace s vedoucím práce bylo rozhodnuto, že bude nutné využít jiné open-source řešení.

4.4 ARCORE

Podle požadavků na vyvíjenou aplikaci vyšel z analýzy knihoven na integraci AR nejlépe framework ARCore od společnosti Google. Jedná se o robustní, optimalizovanou a moderní SDK. Je distribuována jako open source s širokou komunitou vývojářů a kvalitní dokumentací. V následující kapitole 5 Google ARCore je technologie blíže představena.

5 GOOGLE ARCORE

Počátky ARCore vychází z projektu Tango, který byl představen v roce 2014 firmou Google. Jednalo se prvotní úsilí společnosti přivést rozšířenou realitu do mobilního zařízení. Tento nástroj umožňoval vytvořit virtuální trojrozměrnou představu snímaného prostoru. Zařízení, na kterém byla aplikace nasazena, však muselo disponovat hardwarovým hloubkovým senzorem. Zjistilo se, že se nejedná o příliš uživatelsky přívětivý způsob práce s VR, protože bylo jen minimum zařízení, která toto kritérium dokázala splnit. Projekt postupem času skončil s neúspěchem a byl ukončen.

Společnost Google se proto rozhodla vyvinout platformu, která bude více dostupná širší veřejnosti a bude využívat pouze dostupný hardware běžného mobilního zařízení a vytvořila platformu ARCore. Technologie ARCore byla představena v březnu roku 2018 a je plně podporována systémem Android od verze 7 (Nougat) nebo vyšší. ARCore SDK je volně dostupný a je šířen pod licencí GNO. Je neustále vyvíjen a s každým novým vydáním přináší nové a vylepšené funkce rozpoznávání. Momentálně společnost Google pracuje na ARCore Depth API, které rozšiřuje původní rozhraní o propracovanou práci s okolím pomocí teplotních RGB map. Vznikne tedy možnost softwarové práce s hloubkou obrazu bez hardwarových komponent. [12]

Obr. 5-1 ARCore logo [12]

Pro integraci virtuální reality do reálné scény využívá ARCore tři klíčové technologie:

- snímání pohybu (Motion Tracking);
- porozumění prostředí (Enviromental Understanding);
- odhad světla (Light Estimation).

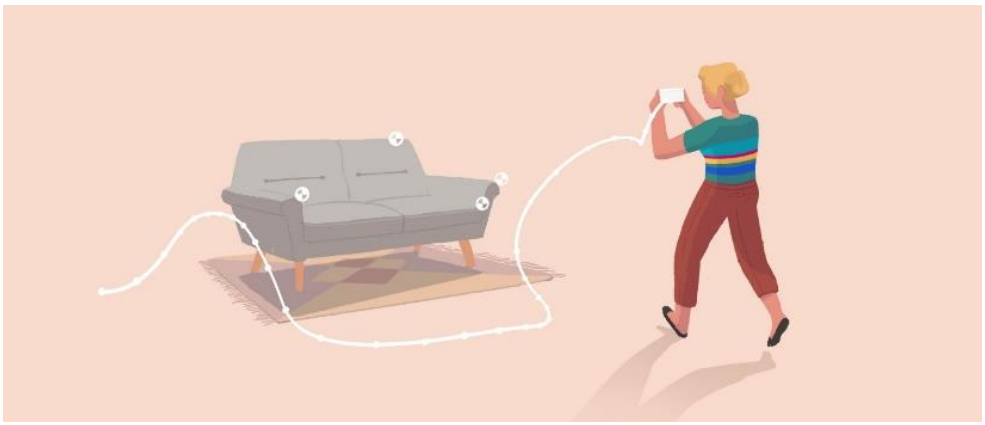
5.1 SNÍMÁNÍ POHYBU

Ke správnému umístění virtuálního objektu do snímané scény musí zobrazovací zařízení přesně vědět svoji aktuální polohu vůči okolnímu prostředí. Podle dokumentace [12] společnost Google k získávání této informace vyvinula a patentovala vlastní algoritmus Souběžné odometrie a mapování (Concurrent Odometry and Mapping, dále COM) [13], který má kořeny v metodě Současné lokalizace a mapování (Simultaneous localization and mapping, dále SLAM). Jedná se o proces, kdy zařízení kontinuálně vytváří trojrozměrnou mapu snímaného prostředí, kterou zároveň využívá k určování své aktuální pozice a orientace. [14]

V metodě COM jsou ve snímané scéně pomocí detektoru detekovány významné body (feature points), které slouží v kombinaci s daty z IMU jako vstupní základ pro algoritmus, viz Obr. 5-2.

Pojem „významný bod“ označuje místo v obraze, které má tyto vlastnosti:

- jasně definovaná pozice v obrazovém prostoru;
- matematická definice;
- lokální struktura okolo významného bohu je bohatá na informace pro následné zpracování významného bodu;
- především je stabilní z hlediska vlivu lokálních nebo globálních deformací v obrazové doméně;
- musí být zaručeno znovunalezení bodu s vysokým stupněm opakovatelnosti.



Obr. 5-2 Ilustrace vyhledávání významných bodů a Motion tracking [12]

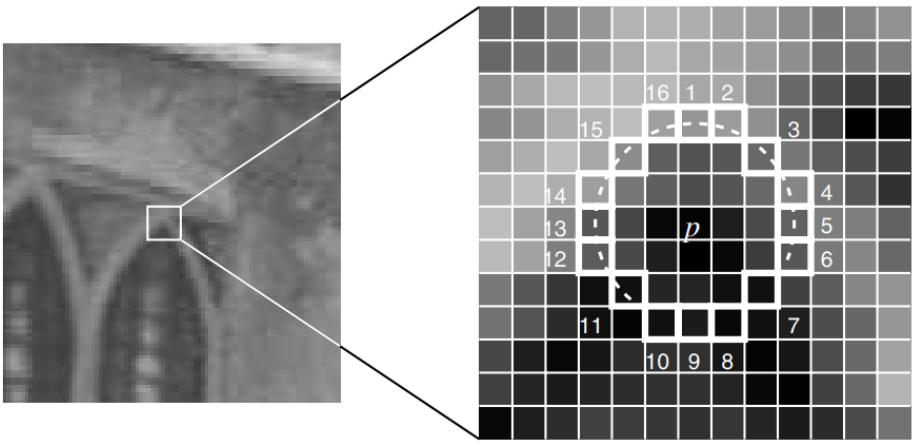
V tomto kontextu jsou významné body především vrcholy nebo hranice objektů, viz Obr. 5-3. Velmi důležitou vlastností kvalitního významného bodu je právě opakovatelnost, která koreluje se spolehlivostí. Detektor musí být schopen již jednou zaznamenaný významný bod znovu nalézt i po působení geometrických a fotometrických změn. Vliv mohou mít například osvětlení, úhel pohledu, perspektiva, rotace, zvětšení nebo rozmazání obrazu během pohybu. Poté, co jsou významné body detekovány, je pro každý významný bod spočítán deskriptor. Prostřednictvím deskriptoru je možné identifikovat body zájmu z různých pohledů a natočení snímacího zařízení.

Obr. 5-3 Zobrazení sledovaných významných bodů ARCore

Není oficiálně uvedeno, jaký detektor a deskriptor bodů zájmu v obrazu je v ARCore využit. Mezi nejpravděpodobnějšími kandidáty je v několika zdrojích uveden deskriptor *BRISK* (Binary Robust Invariant Scalable Keypoints), který využívá modifikovaný detektor FAST (Features From Accelerated Segment Test) [15] [16].

FAST

Features From Accelerated Segment Test neboli FAST, navrhnutý Edwardem Rostenem a Tomem Drummondem v roce 2006, je jeden z nejrychlejších algoritmů pro detekci významných bodů. Principem detektoru je srovnání intenzity I_p vybraného pixelu p s pixely ležícími na kružnici ve vzdálenosti r od tohoto středového pixelu p. Jako první jsou vyhodnoceny hodnoty jasu pro pixely 1, 5, 9, 13, viz Obr. 5-4. Pokud by měl být vybraný pixel rohový, musí být alespoň tři pixely z těchto čtyř porovnávaných s hodnotou intenzity jasu nižší než $I_p - t$ nebo vyšší než $I_p + t$, kde t představuje hodnotu prahu. Pokud je tato podmínka splněna, pixel je vybrán jako potenciální příznak. Pokud není podmínka splněna, je tento bod vyřazen. Následně jsou validovány další pixely na kružnici. V případě, že existuje kolekce n pixelů v řadě (v případě obrázku 12 pixelů) s vyšší nebo nižší intenzitou jasu, jak je zobrazeno na Obr. 5-4 čárkovaně, je toto místo vybráno jako bod zájmu. V závislosti na délce řady splňující tuto podmínku existuje více variant deskriptoru FAST. [17]



Obr. 5-4 Popis rohového detektoru FAST [17]

BRISK

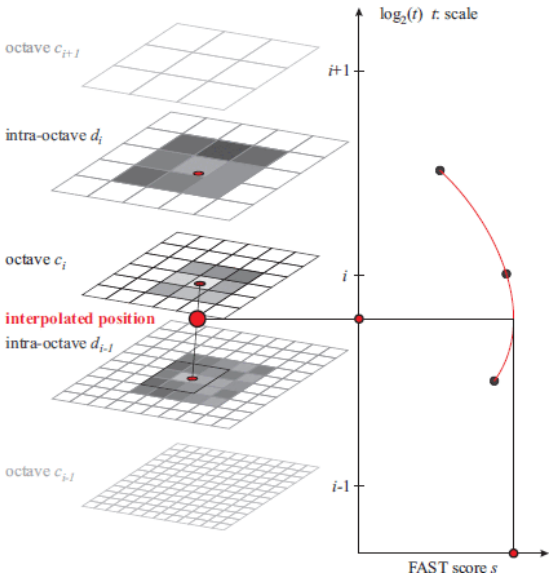
Binary Robust Invariant Scalable Keypoints neboli BRISK je binární deskriptor k detekci rohů, který je invariantní vůči rotaci a změně měřítka. V této metodě jsou detektory porovnávány pomocí bitových operací. Publikován byl autory Stefanem Leuteneggerem, Margaritou Chli a Rolandem Siegwartem v roce 2011 na univerzitě ETH v Curychu. [15]

Deskriptor představuje strukturu, která popisuje vlastnosti charakterizující významný bod, dovoluje tento bod znovu nalézt i po působení geometrických a fotometrických změn.

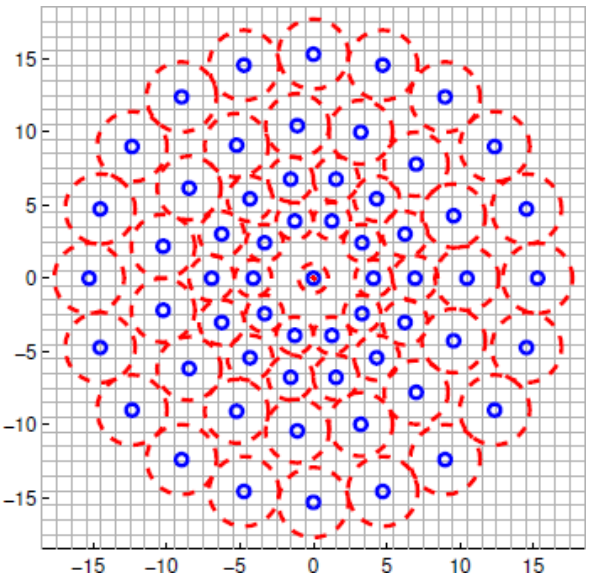
Princip fungování deskriptoru je rozdělen do následujících dvou kroků.

- V prvním kroku dochází k detekci významných bodů metodou FAST. Měřítkový prostor zde tvoří pyramida a je v oktávových vrstvách kvůli omezení vlivu velikosti. Odhad skutečného měřítka je pomoci interpolace, viz Obr. 5-5.
- Druhým krokem po vytvoření sady příznaků je vytvoření binárního popisu významného bodu. BRISK deskriptor tvoří binární řetězec. Ke každému zkoumanému významnému bodu je přiřazen škálovatelný kruh o určitém měřítku N, se středem odpovídajícím významnému bodu. Okolí zkoumaného bodu je poté tímto kruhem vzorkováno, viz. Obr. 5-6. Za účelem odstranění šumu okolí je použito Gaussovo jádro. Prostřednictvím intenzit lokálních gradientů je stanovena charakteristická orientace bodu, a poté jsou orientované vzory využity pro získání výsledných párových porovnání světlosti. Sloučením poté vzniká popis klíčového bodu.

Zdroj udává, že použitím modifikované metody FAST v algoritmu BRISK, lze získat podobné kvalitativně srovnatelné výsledky jako při použití metody SURF (Speeded up robust features), avšak výpočetní náročnost je výrazně nižší. BRISK v kombinaci s FAST představuje rychlou a efektivní metodu použitelnou na mobilním zařízení v AR aplikaci, která požaduje odezvu v reálném čase. [18] [15]



Obr. 5-5 Měřítkový prostor a interpolace měřítka [15]

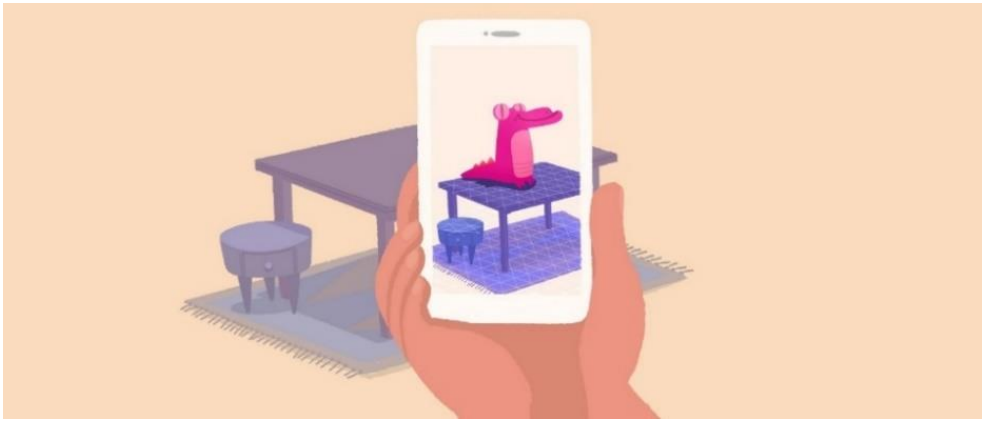


Obr. 5-6 Vzorkování okolí významného bodu [15]

5.2 POROZUMĚNÍ OKOLNÍHO PROSTŘEDÍ

Neustálým snímáním scény se vytváří stále přesnější virtuální model reálného světa pomocí sledování bodů zájmu. ARCore skenováním okolí detekuje shluky významných bodů k odhalení a následnému odhadnutí ploch v reálném prostoru. U ARCore jsou shluky pomocí konvexních polygonů převedeny na horizontální, vertikální nebo nakloněné roviny. Na základě těchto faktů je velice důležité, aby snímaná plocha měla texturu, která obsahuje dostatečné množství významných bodů, což nese jistá omezení. Při snímání scény musí být zajištěn dostatek světla bez reflexí. Nemělo by docházet k rychlému pohybu kamery, který vede k rozmazání obrazu. Poté lze s relativně velkou přesností (na jednotky milimetrů) získat měřítko reálného světa, které je aplikovatelné i na virtuální objekty.

Plochy se využívají v AR aplikaci k uchycení pomocí kotev, a k umisťování virtuálních objektů do scény, viz Obr. 5-7. Detekované plochy, významné body a kotvy jsou v API souhrnně označovány jako *Trackables*. [12]

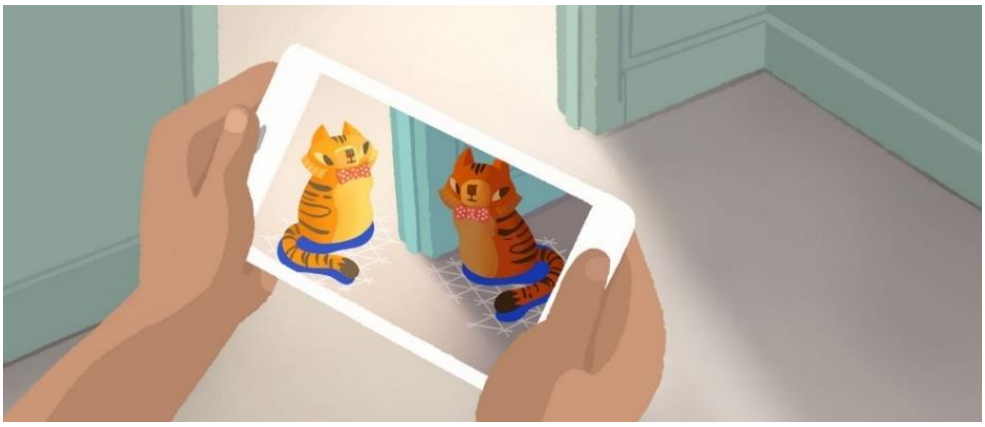


Obr. 5-7 Ilustrace porozumění okolního prostředí a umístění virtuálního objektu [12]

5.3 ESTIMACE SVĚTLA

K zajištění co největší věrohodnosti, realistického podání virtuálních objektů a prohloubení AR zážitku uživatele podporuje ARCore API odhad světelných podmínek okolního prostředí. Použití této funkce v projektu lze využít při vykreslování virtuálních objektů. Zjištěním barvy zdroje světla, jeho intenzity a směru lze tuto informaci objektům

předat a nastavit jejich osvětlení stejně, jako je osvětlení snímané scény. V neposlední řadě, jak je ilustrováno na Obr. 5-8, lze přizpůsobit stín a reflexi vrhanou objektem v závislosti na směru zdroje světla. [12]



Obr. 5-8 Ilustrace estimace světla na základě zdroje světla [12]

6 ANALÝZA VÝVOJOVÝCH NÁSTROJŮ PRO AR APLIKACE

K vývoji mobilní aplikace byl vybrán herní engine. V této kapitole je vysvětlen důvod použití této technologie a vytvořena analýza tří herních enginů, které by bylo možné k implementaci použít. Na základě tohoto souhrnu byl poté vybrán nejvhodnější kandidát.

6.1 HERNÍ ENGINE

Herní engine je softwarový framework navržený především k vývoji videoher a tvoří jádro celé hry, je však možné ho využít i při tvorbě AR aplikací. Stará se o veškerou fyziku, načítání a rendering grafiky, k detekci a řešení kolizí objektů, spouštění animací, přehrávání zvuku a mnoho dalšího. Tento nástroj významně urychluje celý proces tvorby. Odstiňuje vývojáře her od nízko-úrovňových aspektů, čímž umožňuje se více soustředit na samotnou logiku a obsah produktu. Velkou výhodou enginu je multiplatformnost. Stejný framework lze využít u různých typům produktů a usnadňuje následné portování softwaru na odlišné platformy.

Na trhu je k dispozici celá řada ověřených herních enginů a některá vývojářská studia si z důvodu velmi specifických nároků na své produkty vyvíjí i vlastní. Do hlavní množiny nejoblíbenějších volně dostupných enginů patří CryEngine, Unreal Engine a Unity3D. Jednotlivé enginy mohou přistupovat ke stejné problematice různým způsobem, při výběru tedy záleží na konkrétních požadavcích vyvíjeného produktu.

6.1.1 CryEngine

Za vývojem CryEngine stojí německá firma Crytek, která jej poprvé představila v roce 2002. O rok později byl prezentován ve hře Far Cry. Uživateli nabízí ucelený nástroj k vývoji her. K plnému využití tohoto engine je nutné mít již dostatek zkušeností s vývojem. Umožňuje tvorbu her s velkým herním světem, disponuje propracovaným nástrojem na tvorbu terénů a celého prostředí. Programovací jazyk určený k psaní skriptů je C++. Engine umožnuje vývoj her na platformy Playstation, Xbox, iOS, Android, Windows a Linux. [19]

6.1.2 Unreal Engine 4

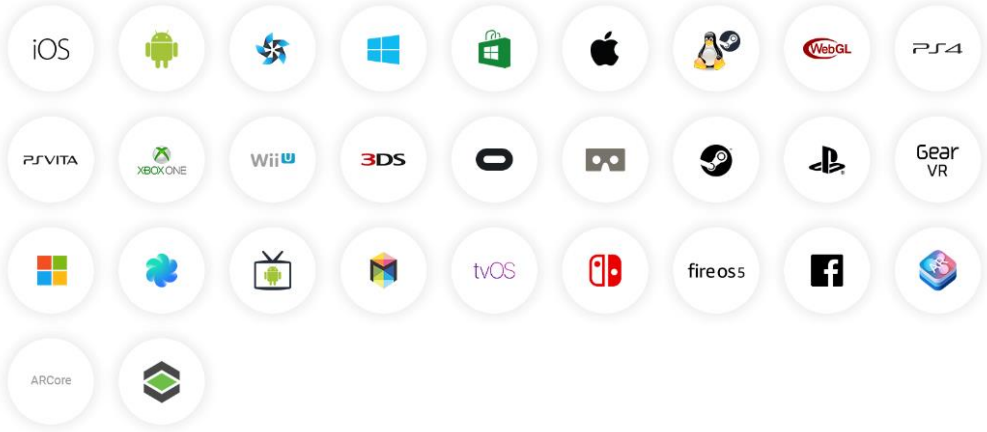
Unreal Engine je jedním z nejotevřenějších a nejpokročilejších real-time 3D enginů. Byl vydán firmou Epic Games v roce 1996 a poprvé byl použit v roce 1998 ve hře Unreal. Původním záměrem bylo využít engine především u akčních her z pohledu první osoby, ale postupně se rozšířil i do ostatních žánrů. Hlavní skriptovací jazyk je C++. Nyní se jedná o všestranný multiplatformní nástroj, který najde využití v produkci filmu, vývoji her, automotive, trénovacích simulátorech a v mnoho dalším. [20]

6.1.3 Unity3D

Unity3D je herní engine, vyvíjený společností Unity Technologies, určený k vývoji 2D a 3D videoher, k produkci filmů a animací, k návrhu architektonických staveb, konstrukčních modelů či simulací a v neposlední řadě k tvorbě grafiky nebo AR a VR aplikací. Podporuje 27 platforem jako jsou počítače, konzole, televize, mobilní nebo AR a VR zařízení s různými operačními systémy viz. Obr. 6-1.

Díky široké komunitě a podpoře ze strany Unity je k dispozici detailní programová dokumentace, včetně propracovaných návodů k většině funkcí Unity. Engine je napsán v jazycích C, C++ a C# a umožňuje vyvíjet nové skripty v jazyce JavaScript nebo C#. Při dodržení licenčních podmínek je možné Unity3D využívat zcela zdarma. [21]

K vývoji mobilní AR aplikace, která je předmětem praktické části, byl použit právě engine Unity ve verzi 5.6. Je ideální pro menší projekty s plnou podporou rozšířené reality pomocí SDK ARCore a operačního systému Android. Nabízí kompletní balík vývojových nástrojů, kvalitní dokumentaci a možnost rychlého ladění aplikace. K psaní skriptů, lze použít programovací jazyk C#, který není náročný na správu paměti.



Obr. 6-1 Podporované platformy Unity 3D

7 UNITY 3D

K vývoji AR aplikace byl použit herní engine Unity 3D. Unity v sobě integruje vlastní nativní vývojové prostředí pro tvorbu 3D a 2D video her nebo aplikací, viz Obr. 7-1. Nabízí se zde nepřeberné množství funkcí, které jsou skvěle vysvětleny v dostupné dokumentaci na webu [22]. Úkolem této diplomové práce není vysvětlení principu Unity, ale jeho správné aplikování při vývoji AR aplikace.

7.1 ZÁKLADNÍ POJMY

K pochopení práce je nezbytné v průběhu celé praktické části vysvětlovat základní pojmy a funkce a s tím spojené názvosloví z Unity. Následující text vychází z volně dostupné dokumentace Unity [22].

Asset

Asset reprezentuje kterýkoliv objekt použitý v projektu. Asset představuje např. herní objekt (G*ameObject*), textury, zvuk, shader, materiál, kameru, scénu, skript, 3D model, obrázek. Asset může být vytvořen přímo v Unity nebo v externím programu a následně importován do projektu. Unity podporuje širokou škálu externích programů na tvorbu Assetů s různými výstupními souborovými formáty.

Unity nabízí Asset Store, což je internetový obchod, kde vývojáři mohou získat zcela zdarma nebo za poplatek nejen obsah pro své projekty, ale i různá rozšíření Unity. Obsah obchodu tvoří především vývojáři z Unity komunity.

GameObject

Základní prvek scény je *GameObject* neboli herní objekt. Každý objekt obsažený ve scéně je typu *GameObject*. Chování a vzhled jednotlivých objektů jsou definovány pomocí komponent, které jsou k nim přiřazeny. Nejzákladnějším je prázdný objekt (*EmptyObject)*, který obsahuje pouze komponentu *Transform* určující pozici, rotaci a velikost objektu. *GameObject* může být použit jako rodič pro několik dalších herních objektů, které společně tvoří stromovou strukturu vyobrazenou v panelu *Hierarchy view*, viz. kapitola Unity IDE.

Komponenta

Vlastnosti objektů určují komponenty, které jsou k nim přiřazeny. Při vytvoření nového objektu ze základní nabídky Unity objektů je ve výchozím stavu k objektu přiřazena kolekce komponent. Základní komponentou, kterou musí obsahovat každý objekt je komponenta *Transform*. Pokud má objekt vykazovat uživatelsky specifickou funkcionalitu, je umožněno pomocí skriptu psaného v jazyku C# vytvořit vlastní s daným chováním. Základní struktura nově založeného skriptu obsahuje dvě klíčové metody:

- Metoda *Start()* je spuštěna pouze jednou při inicializaci objektu, ke kterému je nová komponenta přiřazena.
- Metoda *Update()* je nekonečným cyklem běžícím po celou dobu existence komponenty.

Prefab (prefabrikát)

K přehlednější a strukturalizované správě objektů umožňuje Unity vytvářet tzv. prefabrikát (*prefab)*. Jedná se o vzor objektu, případně skupiny objektů společně tvořících komplexní objekt. Pokud je tento vzor využit ve scéně na vícero místech, v případě potřeby změny vlastností tohoto objektu stačí upravit základní *prefab* šablonu a změna se projeví ve všech instancích objektu v celém projektu.

Scéna

Scéna nabízí prostředí pro objekty aplikace. Scénou v Unity je kolekce Unity objektů, které obsahují své nastavení a skripty. Vyvíjená AR Aplikace se skládá ze dvou scén, které tvoří uživatelské menu a AR prostředí. Scény jsou uloženy v souboru *Assets/Scenes*. V průběhu cyklu aplikace jsou přepínány podle potřeby uživatele pomocí skriptů. K jejich správě za běhu aplikace se v Unity využívá *SceneManager*. Každá scéna, která byla při sestavení aplikace v horní části *Scenes In Build v* okně *Build Settings* zahrnuta do kompilace aplikace, viz. Obr. 2-1, získá svůj identifikační index. S těmito indexy poté pracují metody ve vytvořeném skriptu, kde jsou pomocí funkce *SceneManager.LoadScene(int index)* zmíněné scény vyvolávány. Po spuštění aplikace je automaticky načtena scéna s indexem 0 (*MainMenu*).

Následující kód je ukázkou vytvořeného skriptu obsahující metody na obsluhu scén.

```
1.  using UnityEngine;
2.  using UnityEngine.SceneManagement;
3.
4.  public class MeinMenu : MonoBehaviour
5.  {
6.
7.      public void OpenARBarman()
8.      {
9.          if (!BarmanUtils.missingBarmanRestConfiguration)
10.         {
11.             ConfigPath configObject =
12.             BarmanUtils.FindInActiveObjectByName("SettingsFields")
13.             .GetComponent<ConfigPath>();
14.             BarmanUtils.configPath = configObject.configPath;
15.             SceneManager.LoadScene(1);
16.         }
17.         else
18.         {
19.      Debug.Log("Missing configuration from REST API. Unable to start AR Barman.");
20.         }
21.     }
22.
23.     public void QuitApp()
24.     {
25.         Application.Quit();
26.     }
27.
28.     public void OpenMainMenu()
29.     {
30.         SceneManager.LoadScene(0);
31.     }
32.
33.     public void ResetScene()
34.     {
35.         SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);
36.     }
37. }
```

Výpis 7.1 Obsluha scén pomocí SceneManageru

Kamera

Kamera je objekt, který slouží k definování pohledu na scénu. Pozice kamery určuje úhel a výseč zobrazené oblasti scény. Může být statická nebo dynamická a v závislosti na této vlastnosti se pak mění pohled na scénu.

Osvětlení

Nepostradatelnou součástí scény je definování osvětlení. Osvětlení celé AR aplikaci přidává na autentičnosti. Díky znalosti intenzity, směru a barvě světla lze těmto podmínkám přizpůsobit objekty ve scéně. ARCore poskytuje vlastní řešení při práci s estimací světla, jak je zmíněno v teoretické části.

7.2 UNITY IDE

Vývojové rozhraní se skládá z několika oblastí, které jsou plně uživatelsky přizpůsobitelné a nastavitelné. Následující seznam je vysvětlením pro Obr. 7-1.

Hlavní sekce interface jsou:

A) **Panel nástrojů** – Nabízí funkce pro práci a manipulaci s objekty a pohledem ve scéně. Dále jsou zde ovládací tlačítka ke spuštění, pozastavení a krokování náhledu vytvářené scény. V pravé části se nachází vstup k práci s layouty rozhraní, přístup k Unity uživatelskému účtu a verzovací systém *Unity Collaborate*, který byl při vývoji využíván.

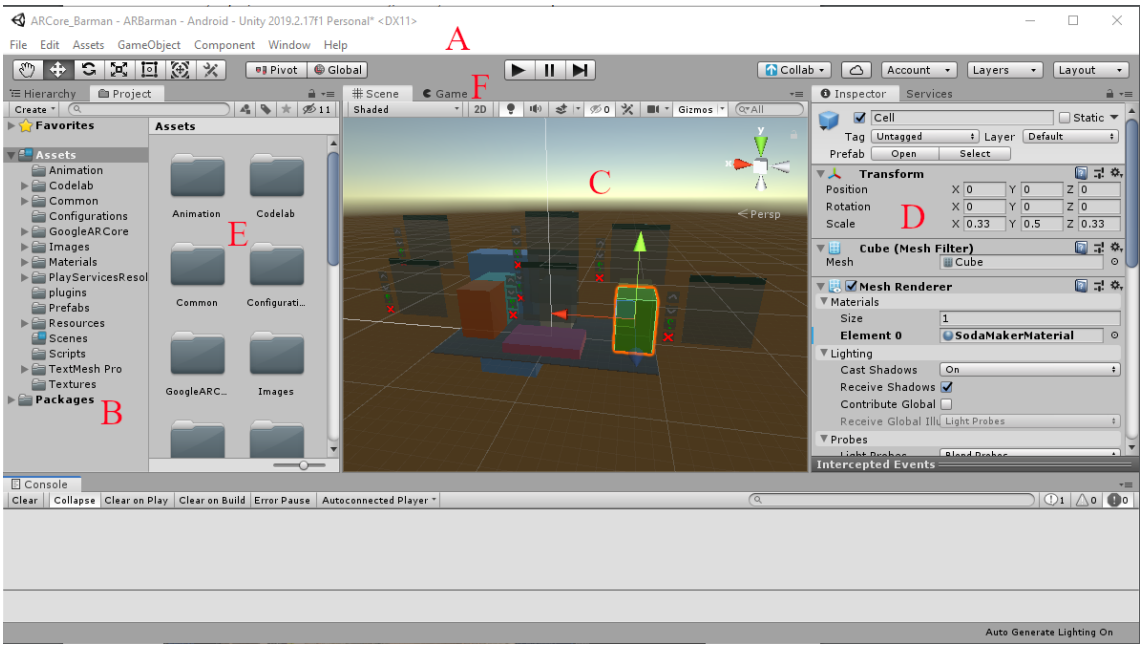
B) **Hierarchy view** – Zobrazuje stromovou strukturu všech použitých objektů, se kterými se v dané scéně pracuje. V této části lze vytvářet a přidávat nové objekty.

C) **Scene view** – Náhled aktuálně načtené scény je jednou z nejdůležitějších částí Unity oken. Zde je tvořen celý obsah aplikace. Aplikaci je možné rozdělit do několika scén a ty během pohybu aplikací postupně načítat. *Scene view* obsahuje všechny objekty, které jsou v *Hierarchy view*. Pohyb po *Scene view* je prováděn prostřednictvím ovládacích prvků v horním panelu nástrojů. Scény jsou ukládány do vlastního adresáře *Assets/Scenes*.

D) **Inspector** – Slouží k zobrazení všech komponent, které jsou přiřazeny k vybranému hernímu objektu. V tomto panelu je umožněno přidat a upravovat již vytvořenou komponentu nebo založit nové pomocí nového skriptu. Následná úprava skriptu probíhá v externím IDE, konkrétně v Microsoft Visual Studio.

E) **Project Assets** – Panel umožňuje zobrazit a editovat všechny Assety, které jsou v projektu k dispozici.

F) **Game view** – Poskytuje vykreslení náhledu kamer ve vytvářené scéně. Zrychluje se tím vývoj, protože není nutné vždy sestavovat aplikaci a testovat ji na koncovém zařízení. Umožnuje širokou škálu nastavení, jako je například výběr z různých poměrů stran zobrazení, a tím simulovat rozdílné displeje. Lze tedy ověřit chování aplikace na rozdílných zařízeních.



Obr. 7-1 Editor Unity – náhled scény

8 IMPLEMENTACE APLIKACE

8.1 POŽADAVKY NA APLIKACI

Cílem mobilní AR aplikace na operační systém Android je integrace rozšířené reality do testbedu Průmysl 4.0. Aplikace umožňuje snímání scény pomocí videokamery zařízení a poté, co je ve scéně detekován testbed, rozšiřuje reálnou scénu o virtuální AR informační systém testbedu. Významem informačního systému je vyobrazovat uživateli informace o aktuálním stavu testbedu pomocí AR. Aktuální informace jsou po aktivaci informačního systému kontinuálně stahovány ze serveru pomocí REST API, které bylo pro testbed v závislosti s touto diplomovou prací vytvořeno. Z REST API je také získávána konfigurace celé aplikace.

Informační systém je tvořen AR informační tabulí, která je přiřazena k buňkám testbedu. Jedná se o:

- buňky umístěné na pracovní ploše
    - zásobník sklenic (Glass Storage);
    - drtič ledu (Ice Crusher);
    - výrobník sody (Soda Maker);
    - homogenizátor nápoje (Shaker);
    - dopravník připravených nápojů (Convoyer);
    - robotický manipulátor SCARA (Manipulator);
- buňku mimo pracovní desku
    - sklad lahví pro přípravu nápojů (Liquor Storage).

Tato AR informační tabule obsahuje nejdůležitější údaje o buňce. Dále by měl informační systém obsahovat i detailní náhled, který je tvořen samostatným oknem.

Je kladen důraz na obecnost a volnost ve formátování zobrazovaných informací. Vzhled prezentovaných dat je řízen ze strany serveru. V případě, že by v budoucnu došlo ke změnám stávajících buněk, musí být umožněno přizpůsobit i prezentaci dat, aniž by musel být proveden zásah do aplikace.

8.2 PRINCIP INTEGRACE AR DO TESTBEDU

Po analýze možných přístupů, jak celou problematiku řešit, vznikl seznam bodů, které je nezbytné splnit k dosažení integrace AR do testbedu.

Jedná se o tyto klíčové kroky:

- získání konfigurace aplikace ze strany serveru;
- zajištění orientace mobilního zařízení v prostoru a s tím spojená detekce pracovní plochy a jednotlivých buněk testbedu;
- trasování pohybů mobilního zařízení vůči testbedu;
- vytvoření virtuálního modelu testbedu a následné aplikování na reálný testbed;
- vytvoření interaktivního AR informačního systému;
- kontinuální získávání dat pro informační systém z REST API testbedu.

V aplikaci je využit framework ARCore k nepřetržitému vytváření virtuálního modelu snímaného prostředí. Tento model je tvořen za podpory dat z IMU v kombinaci s informacemi z videokamery zařízení. Vyvinutá AR aplikace využívá přístup marker-based rozšířené reality. Snímáním pracovní desky robota, na které je umístěn marker, technologie ARCore detekuje plochu jako horizontální rovinu. Tato rovina slouží jako základna pro vytvořený virtuální model testbedu v prostředí Unity. Aplikováním modelu na reálnou scénu v životní velikosti je možné dosáhnout interaktivity AR aplikace s reálným testbedem. V následujících kapitolách je popsán postup implementace a jsou vysvětleny jednotlivé technologie, které jsou využity. Výše zmíněný přístup a jeho principy jsou detailněji popsány v teoretické části práce.

8.3 PŘÍPRAVA ZÁZEMÍ PRO ARCORE V UNITY 3D

8.3.1 Vývojové prostředí

Jako první je nutné získat Unity Editor společně se správcem projektů a instalací *Unity Hub* z oficiálních stránek Unity. Během práce na aplikaci byla použita verze Unity 2019.2.17f. https://unity3d.com/get-unity/download. Všechny nástroje k vývoji AR aplikace popsané níže jsou následně aplikovány v Unity IDE.

ARCore SDK pro Unity

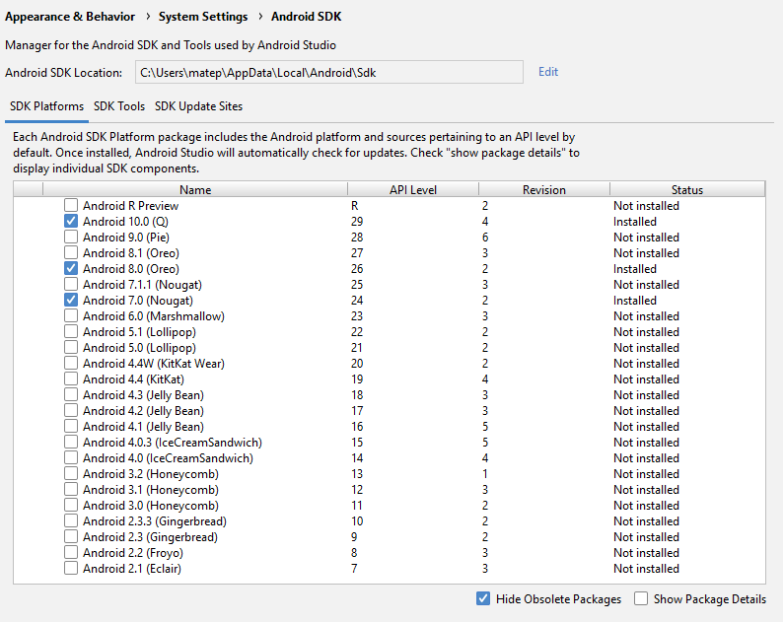
Po založení projektu je dalším krokem import ARCore SDK do Unity prostředí. Import funkcí a *Assetů* je proveden pomocí souborů ve formátu *.unitypackage*. SDK ARCore je získán z GitHub repositáře projektu.

Odkaz na repositář:

https://github.com/google-ar/arcore-unity-sdk/releases odkud je získán soubor *arcore-unity-sdk-1.14.0.unitypackage* .

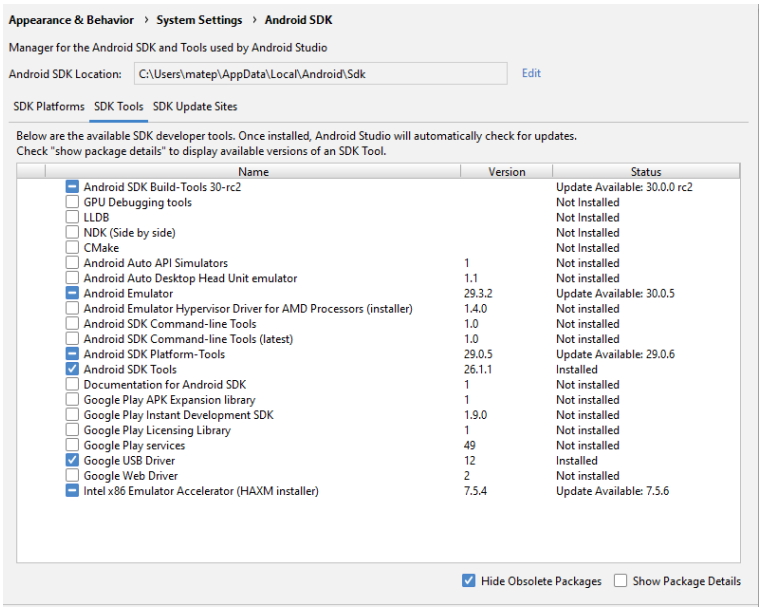
Android SDK Manager

K vývoji aplikace na Android je nepostradatelná přítomnost Android SDK. Ke správě verzí SDK, ovladačů a podpůrných nástrojů k vývoji je možné využít SDK manager z Android Studia. Při vývoji je použito SDK API level 24, avšak jsou k dispozici i vyšší verze, jak je patrné v záložce SDK Platforms, viz. Obr. 8-1. Verze SDK je závislá na ARCore SDK, přičemž nejnižší možná podporovaná verze Android je Android 7.0 Nugat.



Obr. 8-1 Správa Android SDK pomocí Android Manageru

K navázání komunikace telefonu s OS Android a Unity musí být instalován Google USB driver. K rychlejšímu vývoji a ladění aplikace byl použit Android emulátor. Všechny tyto nástroje je možné také spravovat přes Android Manager v záložce *SDK Tools*, viz. Obr. 8-2.



Obr. 8-2 Android Manager správa vývojových nástrojů

Java JDK 11

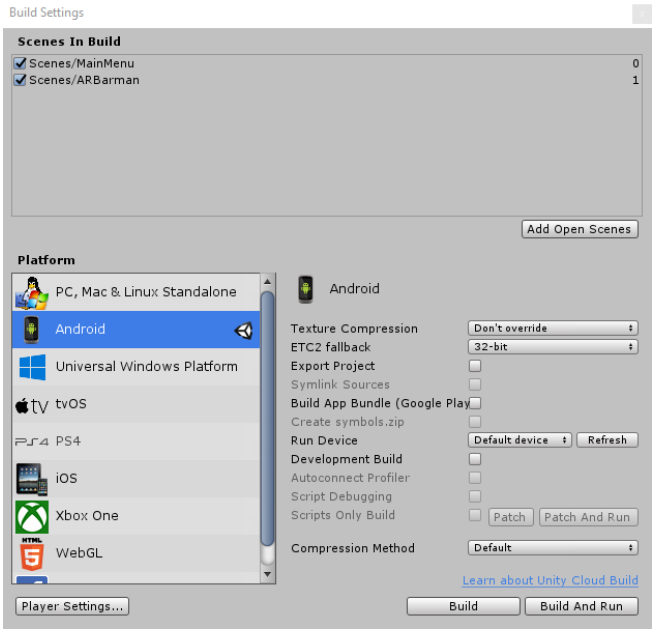
Nezbytná součást vývoje je získání Java SE Development Kit z oficiálních stránek Oracle a následná instalace. Při vývoji je použita konkrétně verze Java SE Development Kit 11.0.5 x64.

8.3.2 Konfigurace projektu

Unity podporuje multiplatformní sestavení vyvíjené aplikace. V případě této diplomové práce je aplikace použita na OS Android. V okně *Build Settings*, které je dostupné z hlavního panelu nástrojů v záložce *File* > *Build Settings*, je dostupná možnost přepínání cílové platformy, viz Obr. 8-3.

V tomto nastavení je nutné před sestavením aplikace vybrat všechny scény, které mají být ve výsledné aplikaci zahrnuty. Každá scéna má přiřazený identifikační index.

Pod tlačítkem „Player settings…“ lze nastavit několik dalších parametrů aplikace. Nutností je zapnutí podpory ARCore v *Player Settings* > *XR Settings* > *ARCore Supported*, dále nastavení minimální API level v *Player Settings* > *Other Settings* > *Minimum API Level* na Android 7 (API Level 24 nebo vyšší). Nesmí se zapomenout na přiřazení odpovídajícího Android SDK v nastavení Unity IDE v záložce *Edit -> External Tools*.



Obr. 8-3 Přepínání mezi cílovou platformou a zahrnutí scén do sestavené aplikace

8.3.3 Příprava zařízení s OS Android

K ladění aplikace během vývoje byl použit smartphone Xiaomi MI9 SE s OS Android 9 s nadstavbou MIUI 11. Před použitím je nutné provést nastavení OS k instalaci aplikací z neznámých zdrojů a Ladění skrz USB. První krok je odemknutí telefonu do vývojářského módu. V případě OS MIUI je postup následující *Nastavení > O telefonu > pětkrát kliknout na MIUI OS verzi systému*. Zobrazí se hláška o získání práv vývojáře. Tímto se uživateli zpřístupní nová sekce v *Nastavení > Další nastavení > Vývojářské nastavení*. V této části lze povolit *USB debugging* a *Install vis USB.*

Služby Google Play pro RR

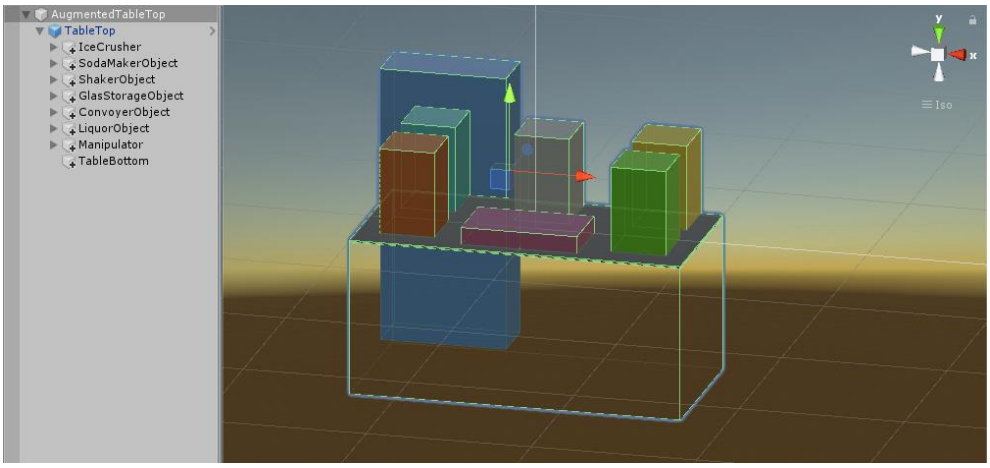
Zařízení, na kterém má být spuštěna AR aplikace využívající ARCore, musí mít nainstalované Služby Google Play pro RR. Jedná se o doplněk pro správné využívání rozšířené reality pomocí frameworku ARCore na mobilním zařízení. Tento doplněk je volně dostupný v obchodu Google Play. Při spuštění aplikace je provedena kontrola přítomnosti služeb Google Play pro RR v zařízení. Pokud nejsou nainstalovány, je uživatel přesměrován na stránku obchodu Google Play, kde je mu umožněno služby nainstalovat. Pokud není požadovaná akce provedena, není možné aplikaci spustit.

Odkaz na doplněk:

https://play.google.com/store/apps/details?id=com.google.ar.core&hl=cs

8.4 VIRTUÁLNÍ MODEL TESTBEDU

K vykreslení AR informačního systému na správných pozicích vůči buňkám testbedu a zajištění kompletní interaktivity aplikace byl v prostředí Unity vytvořen virtuální model testbedu se všemi buňkami. Myšlenkou využití tohoto modelu v AR aplikaci je model aplikovat v životní velikosti na skutečný testbed po detekci umístění testbedu v reálném prostoru. Díky tomu je aplikaci umožněna interakce s reálným testbedem. Virtuální model testbedu se skládá z jednoduchých geometrických objektů, viz. Obr. 8-4. Jednotlivé objekty slouží k vymezení prostoru, který zaujímá jeho skutečný protějšek. Pokud uživatel skrze aplikaci klikne na objekt reálného testbedu, je možné tuto akci detekovat a určit, které části skutečného testbedu se to týká.



Obr. 8-4 Virtuální model testbedu Průmysl 4.0 v prostředí Unity

Virtuální model se skládá z:

- pracovní plochy testbedu;
- spodní části stolu testbedu;
- buněk umístěných na pracovní ploše:
    - oranžová – zásobník sklenic (Glass Storage);
    - tyrkysová – drtič ledu (Ice Crusher);
    - zelená – výrobník sody (Soda Maker);
    - žlutá – homogenizátor nápoje (Shaker);
    - růžová – dopravník připravených nápojů (Convoyer);
    - šedivá – robotický manipulátor SCARA (Manipulator);
- buňky mimo pracovní desku:
    - modrá – sklad lahví pro přípravu nápojů (Liquor storage).

Každá buňka má svoji výchozí pozici. S použitím konfiguračního souboru, který je získán z REST API, lze tuto pozici změnit, viz. kapitola Konfigurační data buněk a markeru. Tato funkcionalita je zde z důvodu možných pozdějších úprav testbedu a promítnutí těchto změn do virtuálního modelu. Aplikaci je tedy možné i nadále používat bez větších zásahů.

K testování a vývoji aplikace disponuje povrch každé buňky barevnou texturou/materiálem. Pokud by buňky zůstaly takto barevné, ve virtuální scéně by kompletně překryly reálné objekty. Při finálním nasazení aplikace je povrch buněk nastaven z konfiguračního souboru z REST API na materiál bez textury. Lze tedy povrch buněk dynamicky měnit na transparentní nebo barevný. Této funkce lze využít při konfiguraci pozic buněk.

U transparentní textury buněk vzniká zásadní problém. K docílení autentičnosti spojení AR a reálné scény musí daná buňka stále vykazovat vlastnosti překryvu okolí. Pokud by neměla žádnou texturu, nebylo by možné při daném úhlu pohledu překrýt např. informační tabuli, která je v zorném úhlu. Toto je vyřešeno pomocí přidání neviditelného materiálu v kombinaci s komponentou *Shader*, který má vlastnost překrýt pozorovaný objekt. Funkcionalita je demonstrována na Obr. 8-5, kde dochází k překryvu neviditelného objektu spodní části stolu s buňkou skladu lahví, která se z daného úhlu pohledu nachází za objektem stolu.

Obr. 8-5 Demonstrace překryvu spodní části stolu s buňkou Sklad lahví

8.5 AR INFORMAČNÍ SYSTÉM TESTBEDU

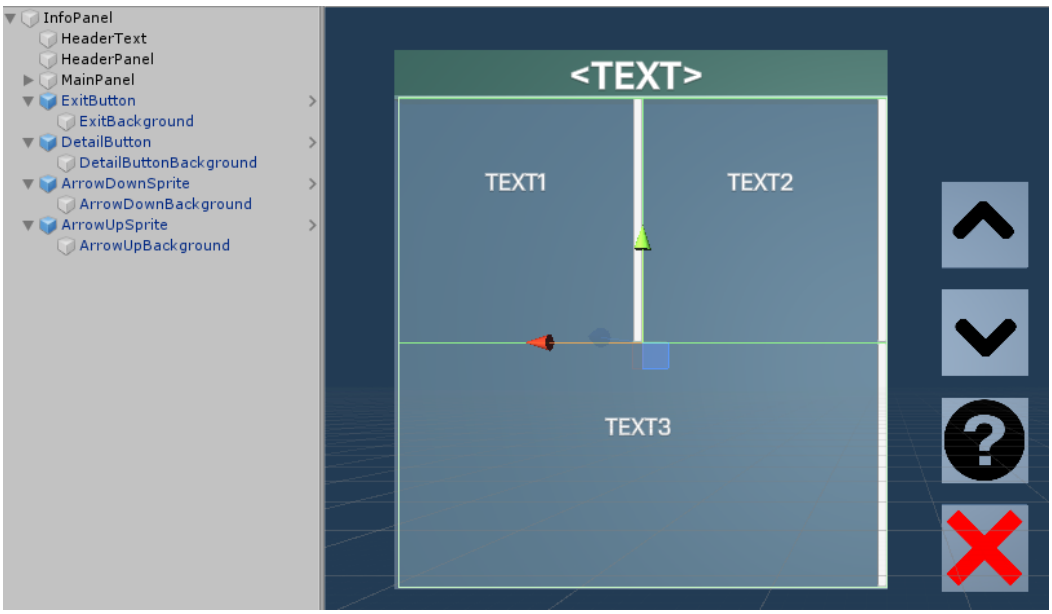
Integrací AR do testbedu vzniká AR informační systém. Účelem systému je prezentace aktuálních dat a aktuálního stavu testbedu v reálném čase. Na základě znalosti pozice testbedu ve snímané scéně reálného prostředí a využití virtuálního modelu, lze vytvářet interaktivní informační tabule ke konkrétním objektům testbedu. Tyto tabule pak uživateli prezentují aktuální stav objektu, ke kterému jsou přiřazeny.

8.5.1 AR informační tabule

Informační tabule pro konkrétní buňku je vyvolána uživatelskou akcí, a to kliknutím na danou buňku včetně Manipulátoru a Dopravníku. Všechny tyto buňky mají přiřazenou komponentu *Box Collider*. Tato kolizní komponenta vytyčuje hernímu objektu tvar pro detekci kolize. Kliknutím na jeden z objektů je aktivován skript *SetVisibleInfoTable* na vytvoření instance informační tabule z prefabrikátu *InfoPanel*. Tabule je zobrazena vedle zvolené buňky.

Informační tabule, viz. Obr. 8-6, je tvořena ze dvou sekcí:

- panel s názvem buňky a texty prezentující data o stavu;
- tlačítka k ovládání informační tabule.



Obr. 8-6 Informační AR tabule

Prezentační sekce

Prezentační část informační tabule obsahuje data o stavu buňky. V hlavičce je uveden název objektu, ke kterému je daná tabule přiřazena. Hlavní panel obsahuje tři posuvná textová pole typu *TextMesh Pro* [23], viz. Obr. 8-6. Obsah polí a jejich velikost jsou řízené pomocí dat získaných z REST API testbedu. Tato data jsou po aktivaci informační tabule kontinuálně získávána ze serveru. Tomuto tématu je věnována vlastní kapitola, viz. REST API.

Textový panel *TextMesh Pro* podporuje formátování textu ve stylu tzv. *Rich Text Format* (RTF). Dalším prvkem, který je využit k zobrazování dat, jsou *Sprites,* neboli 2D obrázky. *TextMesh Pro* umožňuje zobrazování těchto obrázků přímo v textu pomocí tagů s indexem ze Sprite databáze.

Formátování obsahu textových polí pomocí RTF přináší velikou výhodu v řízení vzhledu tabulí vzdáleně bez nutnosti změn v aplikaci. Text obsahuje tagy, které fungují podobně jako například u HTML nebo XML kódu, avšak s méně striktní syntaxí. Následující tabulka obsahuje seznam všech použitelných tagů.

Tabulka 1 Seznam všech tagů RTF u TextMesh Pro

Tag	**Popis**
align	Zarovnání textu
alpha, color	Barva a průhlednost
b, i	Tučně a kurzíva
cspace	Mezery mezi znaky
font	Font
indent	Odsazení
line-height	Výška řádku
line-indent	Odsazení řádku
link	Metadata
lowercase, uppercase, smallcaps	Velká nebo malá písmena
margin	Okraje textu
mark	Zvýraznění textu

Tag	Popis
mspace	Nastavení mezer mezi znaky
noparse	Označení textu jako no-RTF
nobr	Označení nedělitelného textu
page	Konec stránky
pos	Pozice textu na stránce
size	Nastavení velikosti písma
space	Nastavení velikosti mezery
sprite	Vkládání obrázků a animací z databáze Assetů
s, u	Přeškrtnutí a podtržení textu
style	Stylizování textu
sub, sup	Nastavení textu jako horní a dolní index
voffset	Vertikální odsazení
width	Horizontální šířka textu

Tagy jsou zapisovány ve formátu <*tag*>, některé tagy musí být ukončeny pomocí tagu <*/tag*>. Určitým tagům přísluší nastavení hodnoty nebo atributu zápisem <*tag=hodnota*> nebo <*tag attribute=hodnota*>. K detailnějšímu popisu slouží online dostupná dokumentace http://digitalnativestudios.com/textmeshpro/docs/rich-text/.

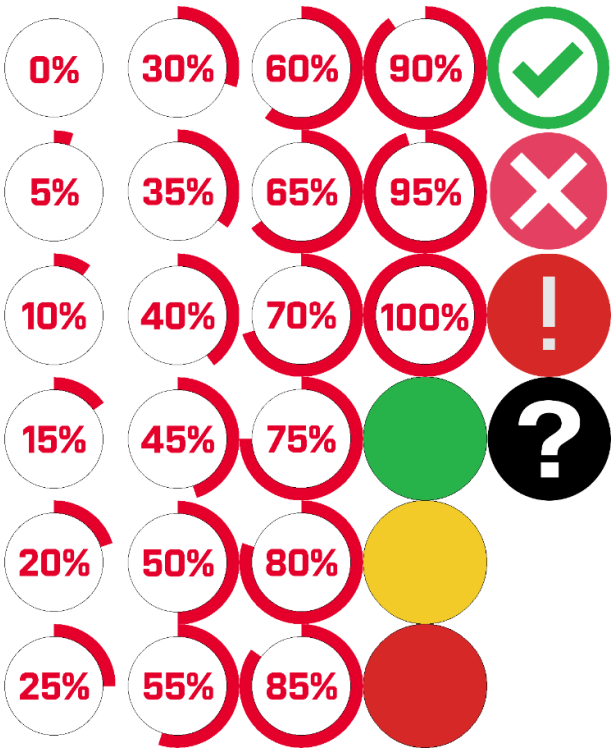
8.5.2 Sprites databáze

Informační tabule používají k přehlednějšímu a intuitivnějšímu zobrazování dat testbedu 2D obrázky, neboli *Sprites.* Textové panely typu *TextMesh Pro* podporují přidávání těchto obrázků do textového obsahu pomocí výše zmíněných tagů.

Sprites jsou sdruženy do kolekce atlasové textury. Pro informační tabule byla ve vektorovém editoru vytvořena textura obsahující 27 obrázků, viz. Obr. 8-7. Tato textura má velikost 1536 x 1280 bodů a je rozdělena na 27 obrázků o velikosti 256x256 bodů.

Obrázky jsou rozděleny do 2 skupin:

- skupina obrázků sloužící jako kruhový ukazatel průběhu (progress bar) s možností zobrazení 0-100 %;
- skupina obrázků k upozornění uživatele.



Obr. 8-7 Textura s obrázky

K vytvoření Assetu se samostatnými obrázky pro komponentu *TextMesh Pro* slouží nástroj *Sprite Editor*. Vstupem je textura v grafickém formátu PNG a popis s rozdělením na samostatné obrázky. Výsledkem je vygenerovaný *Sprite List*, který obsahuje seznam jednotlivých obrázků. Velikost a pivot obrázku je možné v náhledu *Sprite Listu* dodatečně upravit. Pomocí indexu ze *Sprite listu* je možné se odkazovat na obrázek v těle tagu. Indexování začíná od čísla 0. Rozsah indexů je 0-27.

Příklad odkazu na *Sprite* s indexem 2: <sprite=2>, vykreslí obrázek s 10 %.

Ovládací sekce

Ovládací sekce obsahuje čtyři tlačítka určená k ovládání informační tabule, viz. Obr. 8-6:

- tlačítko se znakem šipky vzhůru určeného k přepnutí na následující stránku;
- tlačítko se znakem šipky dolů určeného k přepnutí na předchozí stránku;

- tlačítko se znakem otazníku určeného k otevření nového okna s detailnějším popisem dané buňky, viz. kapitola 8.5.3 Okno detailní popis buňky;
- tlačítko se znakem křížku určeného k uzavření informační tabule a ukončení spojení s REST API.

Zobrazované informace je možné rozdělit na stránky. Při stisknutí tlačítek šipky nahoru nebo dolů je změněn parametr při volání REST API služby. Tím je docíleno změny stránky informační tabule. Obsah stránek a vzhled je řízen ze strany serveru. Pokud se uživatel dostane na poslední dostupnou stránku, je možné šipku vzhůru, pro načtení další stránky, deaktivovat.

Při stisknutí tlačítka s otazníkem se uživateli zobrazí nové samostatné okno, které slouží k detailnímu náhledu k dané buňce, viz. kapitola 8.5.3. Na základě této akce je ukončen cyklus získávání dat z REST API pro základní informační tabuli, a je vytvořeno nové spojení k REST API, avšak s jinou adresou zdroje, určeného pro distribuci dat pro Okno s detailním popisem buňky, viz kapitola 8.5.3.

8.5.3 Okno detailní popis buňky

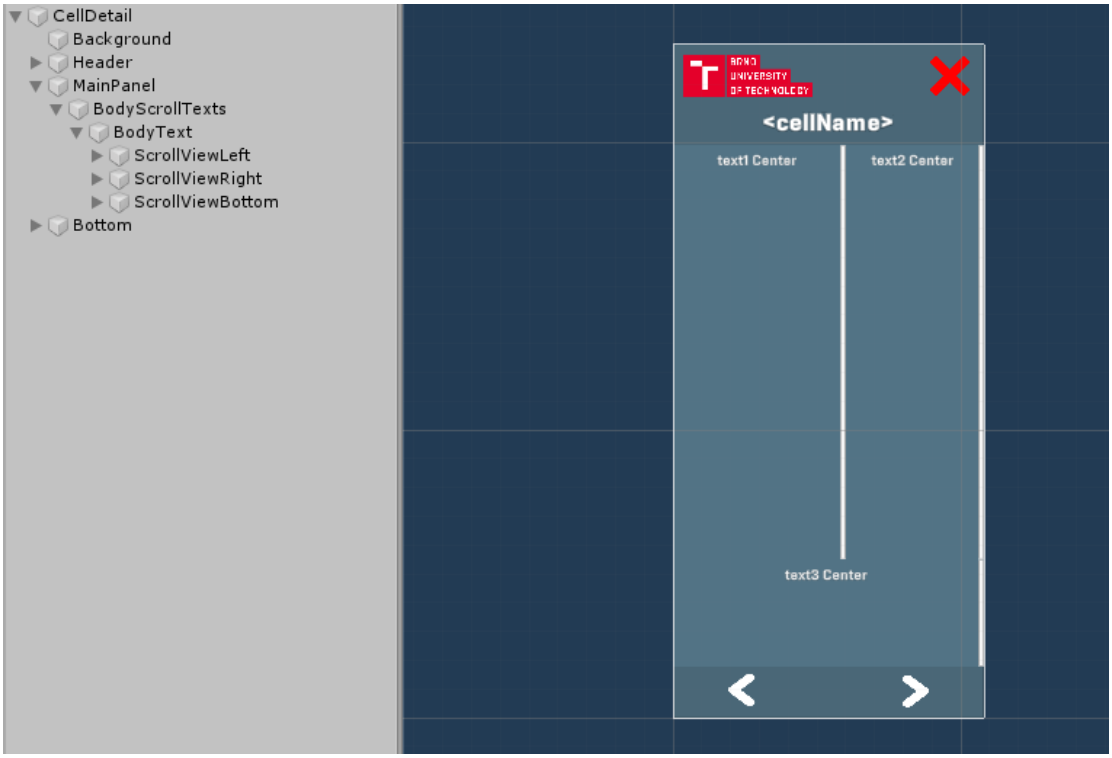
Okno detailního popisu slouží k prezentaci většího množství informací, které mají nižší prioritu důležitosti a nemusí být zobrazovány přímo na základní informační tabuli. Vytvořená instance okna vychází z prefabrikátu zobrazeného na Obr. 8-8.

V hlavičce okna se nachází:

- logo VUT;
- tlačítko se symbolem červeného kříže k uzavření okna a vrácení se zpět na scénu *ARBarman*;
- název buňky, pro kterou je okno otevřené.

Struktura a princip chování prezentační sekce je stejná, jako je to u základní informační tabule. Je rozdělena do tří samostatných textových polí, kde obsah a velikost je řízen z dat získaných z REST API.

Ve spodní části panelu se nachází dvě tlačítka se symboly šipek vlevo/vpravo. Tato tlačítka slouží, stejně jako u základní informační tabule, k přepínání mezi stránkami.



Obr. 8-8 Prefabrikát - Okno detailního popisu buňky

9 ROZHRANÍ APLIKACE

Uživatelské rozhraní aplikace, dále jen UI, je vícejazyčné. Dostupné jsou tři jazykové sady výchozí jazyk angličtina, čeština a němčina. V nastavení aplikace je možné jazyk změnit. Aby aplikace měla sjednocený vizuální styl, jako výchozí písemný font byl zvolen Vafle VUT a základní barva je použita VUT červená RGB (228, 0, 43).

AR aplikace je rozdělena do dvou Unity scén:

- hlavní menu (*MainMenu*);
- AR Barman (*ARBarman*).

9.1 HLAVNÍ MENU

Uživatelské rozhraní hlavního menu tvoří tzv. *Canvas*. Jedná se o *GameObject* s komponentou *Canvas,* který specifikuje oblast, do které jsou objekty (tlačítka, obrázky popisky) UI umístěny. Oblast je definovaná jako rovnoběžník a objekty UI musí být potomky objektu *Canvas*. Vlastností *Canvas* je specifický vykreslovací mód. Existují zde tři módy:

- **Screen Space Overlay** – tento mód umisťuje *Canvas* s UI prvky do přední části scénya vyplní celou výseč pohledu kamery. UI prvky Canvasu vždy překryjí ostatní objekty scény. Tento mód je použit pro hlavní menu.
- **Screen Space Camera –** tento mód objektu *Canvas* je stejný jako Screen Space Overlay s tím rozdílem, že je *Canvas* umístěn s definovanou vzdáleností od kamery.
- **WorldSpace –** *Canvas* se v tomto módu chová jako ostatní objekty ve scéně. *S* určenou velikostí může být do scény umístěn na vybranou pozici. Může být překryt ostatními objekty scény, které jsou v zorném úhlu kamery. Tento mód je použit ve scéně *ARBarman* u informačních tabulí.

Po spuštění aplikace je načtena základní scéna s indexem 0 obsahující hlavní menu, viz. Obr. 9-1, které slouží jako rozcestí do dalších částí aplikace. V této scéně je použit *Canvas* s UI prvky v módu *Screen Space Overlay*. *Canvas* přizpůsobí velikost a umístění UI prvků tak, aby byly správně zobrazeny na všech typech zařízení s rozdílným rozlišením a poměrem stran displeje.

Přechod mezi jednotlivými částmi menu (úvod, nastavení, informace) je proveden prostřednictvím aktivování a deaktivování objektů v jednotlivých částech nabídky. Dané menu je tvořeno stromovou strukturou objektů (tlačítka, logo, texty). Kořenovým rodičem je prázdný *GameObject*. K efektivní práci a rychlejší odezvě uživatelského rozhraní je vždy při přechodu do jiné části menu deaktivován kořenový rodič aktuálně zobrazeného menu, čímž se skryjí i všichni potomci. V návaznosti na tuto událost je aktivován kořenový rodič menu, do kterého se uživatel chce přemístit. Aktivace a deaktivace je provedena pomocí funkce *GameObject.SetActive()* přirazené k jednotlivým tlačítkům v menu. Toto řešení dává rychlejší odezvu uživatelského rozhraní než každému menu tvořit vlastní scénu.

Obr. 9-1 Hlavní menu

Obr. 9-2 Informace o projektu

Obr. 9-3 Nastavení aplikace

Úvodní obrazovka načtená při spuštění aplikace obsahuje viz. Obr. 9-1:

- logo VUT s názvem projektu
- čtyři ovládací tlačítka:
  - AR Barman;
  - Informace;
  - Nastavení;
  - Konec.

9.1.1 AR Barman

Tlačítko AR BARMAN uživatele přesměruje na novou scénu určenou ke snímání testbedu a vytvoření AR informačního systému. Scéna AR Barman je hlavní část aplikace. Spojením frameworku ARCore, dat z IMU v kombinaci s informacemi z video kamery zařízení a dat z REST API testbedu, je v této scéně integrována rozšířená realita do testbedu, čímž vzniká AR informačního systému Barmana. Z důvodu rozsáhlosti funkcionalit je tomuto tématu věnována hlavní kapitola AR Barman.

9.1.2 Informace

Tlačítko INFORMACE zpřístupní část s informacemi o celém projektu testbed Průmysl 4.0 a účelu AR aplikace, viz. Obr. 9-2. Tato nabídka obsahuje rolovací textovou oblast vytvořenou pomocí nástroje *TextMesh Pro* [23].

9.1.3 Nastavení

Nastavení aplikace nabízí dvě funkcionality, viz. Obr. 9-3:

- nastavení jazyka rozhraní aplikace;
- konfigurace REST API URI.

V této části je umožněna uživatelská konfigurace aplikace. Nastavením lze změnit jazyk rozhraní celé aplikace a základ URI adresy pro připojení ke zdrojům REST API testbedu. Další pokročilé nastavení aplikace je řízeno ze strany serveru. Toto je popsáno v kapitole 11.

Konfigurace jazyka

Aplikace disponuje možností nastavení jazyka rozhraní. Na výběr je čeština, angličtina, němčina. Jako výchozí jazyk je angličtina. Pro správu jazyka byl vytvořen skript *Text Manager*. Při spuštění aplikace je v sekci skriptu *Start()* pomocí funkce *TextManager.LoadLanguage()* načten externí textový soubor odpovídající zvolenému jazyku. Soubor obsahuje všechny texty v konkrétním jazyce, je umístěn ve složce *Resources/Languages.* Struktura externího souboru je ve formě mapy. Klíč ke každému prvku mapy je společný v rámci všech jazyků. Tento klíč slouží jako odkaz k nastavení jazyka na daném objektu aplikace, podle kterého je překlad nalezen v načteném zdrojovém souboru aktuálního jazyka rozhraní.

Všechny texty v aplikaci jsou řešeny pomocí nástroje *TextMesh Pro* [23]. K objektu *TextMesh Pro* ve scéně je připojena komponenta *Set Languaged Label*, která pomocí funkce *SetLabelTextInActualLanguage()* a zadaného klíče nastaví odpovídající text.

Konfigurace REST API URI

Nastavení základu URL pro REST API je umožněno změnit v políčku s popiskem REST API. Při otevření okna s nastavením je načtena aktuální URI, Obr. 9-3. Změnu je nutné následně potvrdit tlačítkem ULOŽIT.

9.1.4 Konec

Tlačítko KONEC ukončí celou aplikaci pomocí volání *Application.Quit().*

9.2 AR BARMAN

Scéna AR Barman je jednou z hlavních součástí aplikace. Z důvodu šíře problematiky s ní spojenou je této scéně věnována nová kapitola AR Barman.

10 AR BARMAN

V této scéně dochází k samotné integraci AR do testbedu. V následujícím textu je vysvětlen princip implementované integrace, algoritmus detekce a aplikování virtuálního modelu s AR informačním systémem.

10.1 UŽIVATELSKÉ ROZHRANÍ

Po načtení scény *ARBarman* je aplikací spuštěn fotoaparát zařízení a snímaná reálná scéna je prezentována na displej zařízení. Ve spodní části displeje jsou zobrazena dvě tlačítka Menu a Reset, viz. Obr. 10-2 a Obr. 10-3.

- **Menu** – Po stisknutí tlačítka MENU je uživatel přesměrován zpět do úvodního menu, *SceneManager* načte scénu *MainMenu* s indexem scény 0.
- **Reset –** Tlačítko RESET slouží k znovunačtení scény AR Barman. Lze ho využít v případě problému s detekcí okolního prostředí a aplikování virtuálního modelu testbedu na jeho skutečný protějšek.

10.2 PRINCIP DETEKCE TESBEDU

AR aplikace je založená na principu marker-based AR s frameworkem ARCore. K integraci AR musí být objekt (testbed) reálné scény označen pomocí markeru, prostřednictvím kterého se algoritmus aplikace orientuje v prostoru reálné scény a získá informace k registraci AR. O správu scény se stará herní objekt *SceneController*.

10.2.1 Marker

Marker umístěný na pracovní ploše testbedu má dvě hlavní funkce:

- identifikaci plochy testbedu ve snímané scéně;
- získání potřebných informací jako jsou pozice, rotace, měřítko a vzdálenost plochy vůči snímacímu zařízení pro správnou registraci rozšířené reality.

Jako marker byl vytvořen QR kód. Disponuje rozměry 12 x 12 cm a jeho součástí je logo VUT. Kromě specifické funkce pro rozšířenou realitu slouží i jako odkaz na webové stránky celého projektu www.factory4.eu. Marker je zobrazen na obrázku Obr. 10-1. Pozice markeru na pracovní ploše je přizpůsobitelná. V nabídce nastavení aplikace je uživateli umožněno změnit souřadnice markeru vůči pracovní ploše.

Obr. 10-1 Marker vytvořený pro detekci pracovní plochy testbedu

V horní části displeje je uživateli popisován postup k úspěšné registraci AR. Skládá se ze dvou kroků.

10.2.2 První krok registrace AR

V prvním kroku je nutné naskenovat QR kód umístěný na pracovní ploše testbedu. K tomuto účelu je do scény vložen bílý obrys čtverce, který znázorňuje, jak má být QR kód video kamerou snímán. K této akci je uživatel vyzván větou, NASKENUJ TESTBED QR KÓD, viz Obr. 10-2. QR kód je brán v ARCore API jako tzv. *Augmented Image* [24]. Po detekci je QR kód trasován.

Augmented Image

Jedná se o 2D obrázek, na který lze pomocí ARCore reagovat a v reálné scéně trasovat. Nalezením obrázku lze určit fyzickou polohu, lokaci v reálné scéně a měřítko. Referenční sada obrázků je ARCore předána pomocí databáze obrázků v nastavení ARCore relaci (*ARCore Session*) v herním objektu *ARCore Device*. V této databázi lze předat až 1000 referenčních obrázků a současně lze v jedné scéně trasovat až 20 obrázků. QR kód byl vybrán z důvodu rychlé a přesné detekce.

K úspěšnému a kvalitnímu detekování obrázku musí splňovat tyto podmínky:

- referenční obrázek musí zakrývat nejméně 25 % snímku kamery;
- musí být umístěn na rovný povrch;
- snímání musí být přímé a plynulé, bez reflexí a fotometrických zkreslení obrázku.

10.2.3 Druhý krok registrace AR

V druhém kroku je nezbytné naskenovat část pracovní plochy testbedu, na které se QR kód nachází a následně potvrdit kliknutím na obrázek markeru. Postačí naskenovat nejbližší okolí. K této akci je uživatel vyzván větou „NASKENUJ PLOCHU OKOLO, POTÉ KLINI NA QR KÓD “, viz Obr. 10-3.

ARCore pracovní plochu detekuje jako horizontální rovinu, která slouží jako základ pro virtuální model testbedu. Rozpoznání roviny probíhá pomocí detekce tzv. bodů zájmu (*Features points*). Detailní popis je popsán v teoretické části v kapitole

Porozumění okolního prostředí. Detekovaná rovina (*Detected plane*) v ARCore API spadá pod tzv. *Trackable.* Pojmem *Trackable* je myšlen objekt, který lze s ARCore trasovat a lze k němu připojit kotvu (*Anchor*). Kotvy slouží k upevnění pozice a orientace v reálném prostoru. Díky tomu je možné docílit přesné registrace AR a nedochází příliš k driftu. K potvrzení naskenované okolní plochy musí uživatel kliknout na marker. Ten musí zakrývat alespoň 25 % plochy snímku kamery. Touto akcí je vytvořena kotva připojená k horizontální rovině ve středu referenčního obrázku. Po úspěšné registraci je uživatel informován hláškou. Na základě znalosti pozice, orientace markeru a z toho získaného měřítka, je vytvořena instance virtuálního modelu testbedu v životní velikosti, která je aplikovaná na skutečný testbed.

Obr. 10-2 Scéna ARBarman – první krok při detekci pozice testbedu v prostoru

Obr. 10-3 Scéna ARBarman – druhý krok při detekci pracovní plochy testbedu

10.3 APLIKOVÁNÍ VIRTUÁLNÍHO MODELU TESTBEDU

Se znalostí pozice markeru v reálném prostoru je možné použít virtuální model a aplikovat ho na skutečný testbed. Instance virtuálního testbedu je umístěna na pracovní plochu, která je frameworkem ARCore detekována jako horizontální rovina. K uchycení AR modelu v prostoru je použita kotva (*Anchor*). O správu scény se stará kontrolér scény (*SceneController).*

10.3.1 Kontrolér scény

Úkolem kontroléru je správa registrace AR a navádění uživatele při detekci markeru. Nejdůležitější část skriptu je umístěna v cyklu metody *Update(),* která je spouštěna každý snímek běhu aplikace.

```
1.  if (image.TrackingState == TrackingState.Tracking &&
2.      image.TrackingMethod == AugmentedImageTrackingMethod.FullTracking)
3.  {
4.      TrackableHit hit;
5.      TrackableHitFlags raycastFilter = TrackableHitFlags.PlaneWithinPolygon
6.          | TrackableHitFlags.FeaturePointWithSurfaceNormal;
7.
8.      if (Frame.Raycast(touch.position.x, touch.position.y, raycastFilter, out hit))
9.      {
10.         if (hit.Trackable is DetectedPlane)
11.         {
12.             Debug.Log("Creating table..");
13.             BarmanUtils.FindInActiveObjectByName("ScanOverlay").SetActive(false);
14.             BarmanUtils.FindInActiveObjectByName("TextQRCode").SetActive(false);
15.
16.             DetectedPlane detectedPlane = hit.Trackable as DetectedPlane;
17.             var anchor = detectedPlane.CreateAnchor(image.CenterPose);
18.             SetSelectedPlane(detectedPlane);
19.             tableController.createTable(anchor, image);
20.         }
21.     }
22. }
```

Výpis 10.1 SceneController – vytvoření kotvy a instance virtuálního stolu

V prvním kroku registrace AR, viz. kapitola 10.2.2, kdy je uživatelem naskenován marker, ARCore spouští trasování tohoto obrázku, čímž instance obrázku získává stav *TrackingState.Tracking*. Poté, co je v druhém kroku registrace AR uživatel vyzván ke kliknutí na marker, je v kontroléru při splnění výrazu v if podmínce proveden kód z Výpis 10.1. Dojde k vytvoření kotvy na detekované rovině a následné vytvoření instance a umístění virtuálního modelu do reálné scény. Při umisťování modelu do scény je brán v potaz i offset buněk a markeru, který je brát z nastavení aplikace.

Pokud by došlo ke špatné registraci AR je možné scénu znovu načíst tlačítkem RESET ve spodní části displeje a akci registrace provést znovu.

10.4 AR INFORMAČNÍ SYSTÉM

Pokud je virtuální model správně aplikován na skutečný testbed, je možné kliknutím na buňku skutečného testbedu vykreslit informační tabuli. Obsah informační tabule a tlačítek je popsán v kapitole AR Informační systém . Za inicializaci informační tabule je zodpovědný herní objekt *CellDetailController*. Po vytvoření instance informační tabule je vytvořeno spojení k REST API a dochází ke kontinuálnímu stahování aktuálních dat testbedu. Na Obr. 10-4 a Obr. 10-5 lze vidět informační tabuli a detailní náhled ke konkrétní buňce s načtenými testovacími daty.

Obr. 10-4 Informační tabule s načtenými daty o stavu zvolené buňky (testovací data)

Obr. 10-5 Detailní náhled stavu buňky (testovací data)

11 REST API

Aktuální data o stavu testbedu a konfigurační parametry aplikace jsou získávána z REST API testbedu. REST API bylo vytvořeno v souvislosti s touto diplomovou prací.

11.1 DEFINICE

REST API

REST API, neboli Representational State Transfer Application Programming Interface je architektura pro aplikační programové rozhraní zaměřené na data. REST je použitelný pro jednotný a snadný přístup ke zdrojům (*resources*). Zdroj má svoji unikátní adresu URI a lze k němu přistoupit metodami Create, Retrieve, Update, Delete (CRUD). Přenos dat je možné zprostředkovat vícero způsoby. V případě této práce je proveden protokolem HTTP a to metodami POST, GET, PUT, DELETE. Návratový formát dat je tytu JSON. [25]

JSON

Data jsou přenášena ve formátu *JSON*, neboli *JavaScript Object Notation.* Jedná se o řetězec, kterým je možné zapsat v hierarchickém uspořádání jakýkoliv objekt či pole. Pro práci s *JSON* v Unity je použit nástroj *SimpleJSON* [26].

11.2 ZASÍLÁNÍ POŽADAVKU

Přenos dat je v aplikaci řízen REST API kontrolérem *RestAPIController*. Před samotným voláním HTTP GET metody je nutná autentizace vůči serveru pomocí tokenu. Token je získán od serveru po úspěšně autentizaci uživatelského profilu.

Při inicializaci informační tabule nebo otevření detailního náhledu k dané buňce je volána funkce *CallAPI()* kontroléru. Při tomto volání je zahájen koprogram, který každé dvě sekundy získává data z REST API. Podle aktuálně otevřené informační tabule se liší adresa zdroje, na které je zasílán požadavek na data.

Na následující příkladě bude vysvětlen princip složení cílové adresy zdroje:

http://bb.skupra.cz/cell/cz/shaker/InfoTable/1

Adresa zdroje je rozdělena do několika úrovní:

Úroveň 0 - http://bb.skupra.cz/cell/ - jedná se o základní část URI. V nastavení aplikace je možné ji dynamicky měnit;

Úroveň 1 – …/cz/ - představuje jazyk v jakém mají být data v informačním systému prezentována. Jazyk je zvolen podle aktuálně nastaveného jazyka rozhraní;

Úroveň 2 – …/shaker/ - určuje název buňky;

Úroveň 3 – …/InfoTable/ - definuje typ informační tabule

Dostupné dvě varianty:

- InfoTable = základní informační tabule
- DetailTable – detailní náhled k dané buňce);

Úroveň 4 – …/1 – určení čísla stránky.

Následující tabulka obsahuje seznam všech zdrojů REST API testbedu. Tabulka je rozdělena do dvou skupin:

Tabulka 2 Seznam všech zdrojů REST API

Adresa zdroje s konfigurací	http://bb.skupra.cz/api/configuration
Adresy zdrojů pro základní informační tabule buněk.	http://bb.skupra.cz/cell/cz/IceCrusher/InfoTable/1 http://bb.skupra.cz/cell/cz/SodaMaker/InfoTable/1 http://bb.skupra.cz/cell/cz/Shaker/InfoTable/1 http://bb.skupra.cz/cell/cz/GlassStorage/InfoTable/1 http://bb.skupra.cz/cell/cz/Convoyer/InfoTable/1 http://bb.skupra.cz/cell/cz/LiquorStorage/InfoTable/1 http://bb.skupra.cz/cell/cz/Manipulator/InfoTable/1

Adresy zdrojů pro detailní náhled stavu buňky.	http://bb.skupra.cz/cell/cz/IceCrusher/DetailInfo/1 http://bb.skupra.cz/cell/cz/SodaMaker/DetailInfo/1 http://bb.skupra.cz/cell/cz/Shaker/DetailInfo/1 http://bb.skupra.cz/cell/cz/GlassStorage/DetailInfo/1 http://bb.skupra.cz/cell/cz/Convoyer/DetailInfo/1 http://bb.skupra.cz/cell/cz/LiquorStorage/DetailInfo/1 http://bb.skupra.cz/cell/cz/Manipulator/DetailInfo/1

11.3 STAVOVÁ DATA TESTBEDU

Získaná data jsou ve formátu *JSON*. Pokud nedošlo během přenosu k žádné chybě, jsou data obdržena se stavovým HTTP kódem 200. Přijatá data z *JSON* souboru jsou převedena na instanci třídy *RestResponseData*, viz. Výpis 11.1. Pomocí komponenty *RestResponseViewer* přiřazené k textovým polím a tlačítkům informační tabule, jsou data prezentována, viz. kapitola 8.5. *JSON* soubor má pevně danou strukturu, která je odvozen ze šablony třídy.

```
1. public class RestResponseData
2. {
3.     public string language;
4.     public InfoTablTMPUGUIText text1;
5.     public InfoTablTMPUGUIText text2;
6.     public InfoTablTMPUGUIText text3;
7.     public bool buttonUpPage;
8.     public bool buttonDownPage;
9. }
```

Výpis 11.1 Třída RestResponseData

11.3.1 JSON šablona

Soubor s daty o aktuálním stavu buňky má pevně danou strukturu a vychází ze struktury informační tabule, jak je popsáno v rozboru AR informačního systému, viz kapitola 8.5. Příklad souboru je přiložen v příloze diplomové práce.

Šablona se skládá z:

- **language** – jazyk rozhraní informační tabule;
- **text1, text2, text3** – tři objekty typu text s atributy:
  - **text** – obsah pro komponentu TextMesh Pro informační tabule, viz. kapitola 8.5;
  - **width –** určení šířky komponenty TextMesh Pro. Hodnota je udávaná v procentech vůči šířce celé informační tabule;
  - **high –** určení výšky komponenty TextMesh Pro. Hodnota je udávaná v procentech vůči šířce celé informační tabule;
- **buttonPageUp, buttonPageDown –** boolean hodnota k aktivaci a deaktivaci tlačítek k přepínání stránek informační tabule.

```
{
   "language":"CZ",
   "text1":{
      "text":"[string]",
      "width":[percentValue],
      "high":[percentValue]
   },
   "text2":{
      "text":"[string]",
      "width":[percentValue],
      "high":[percentValue]
   },
   "text3":{
      "text":"[string]",
      "width":[percentValue],
      "high":[percentValue]
   },
   "buttonUpPage":[boolean],
   "buttonUpDown":[boolean]
}
```

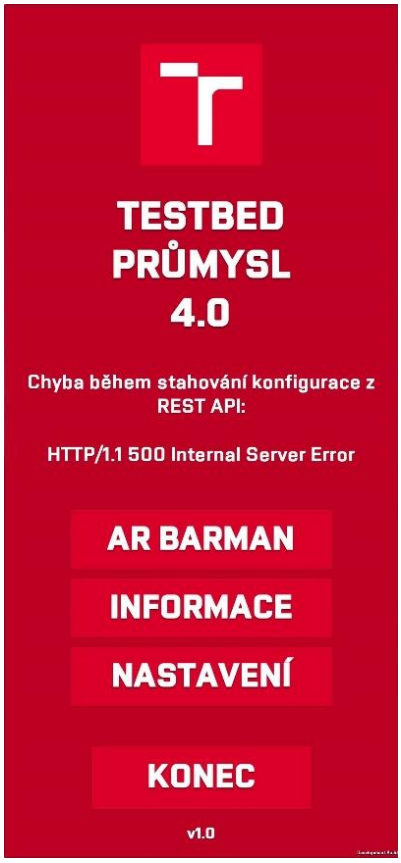
Výpis 11.2 Šablona JSON se stavovými daty

11.4 KONFIGURAČNÍ DATA BUNĚK A MARKERU

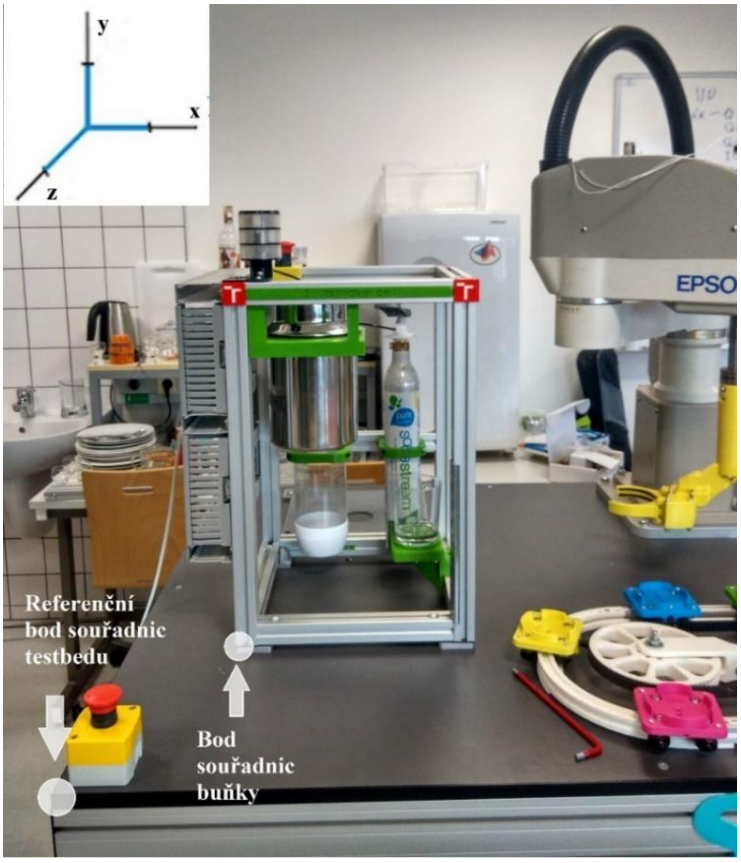
V případě, že by v budoucnu došlo ke změně umístění buněk a markeru na pracovní ploše testbedu, aplikace nabízí možnost kalibrace virtuálního modelu a s tím spojeného AR informačního systému. Při spuštění aplikace je z REST API stažen soubor v JSON formátu s aktuálním nastavením pozic buněk ve virtuálním modelu testbedu. V rámci konfigurace je nutné zadat i pozici markeru na pracovní ploše vůči reálnému testbedu.

Obsah souboru je ve formátu JSON. K následné práci s JSON souborem je použito rozšíření Unity *SimpleJSON* [26].

V případě, že nelze konfigurační soubor při spuštění aplikace načíst, je uživatel upozorněn chybovou hláškou na úvodní stránce. Upozornění odpovídá stavové hlášce, která je získána po chybovém požadavku na server pomocí HTTP volání, viz. Obr. 11-1. V případě úspěšně získané konfigurace není uživatel nijak upozorněn.



Obr. 11-1 Interpretace chybového hlášení při získávání konfigurace



Obr. 11-2 Souřadnicový systém testbedu a buňky

11.4.1 JSON šablona

Soubor s konfiguračními daty má pevně danou strukturu JSON formátu.

JSON soubor se skládá z třech hlavních částí:

- **CellsTransparent** – boolean hodnota *true/false* globálně určující, zda vykreslené buňky virtuálního modelu mají disponovat texturou;

- **Konfigurace buněk** – pro každou virtuální buňku je možné určit pozici vůči reálnému testbedu, název buňky musí přesně odpovídat názvu herního objekt v Unity scéně, příloha diplomové práce obsahuje vzorový JSON soubor;
- **Konfigurace markeru –** nastavení pozice markeru na pracovní desce testbedu.

```
{
  "CellsTransparent": [true/false],
  "[název buňky]": {
    "name": "[název buňky]",
    "x": [float hodnota],
    "y": [float hodnota],
    "z": [float hodnota]
  },
…
…
…
  "Marker": {
    "name": "Marker",
    "x": [float hodnota],
    "y": [float hodnota],
    "z": [float hodnota]
  }
}
```

Výpis 11.3 Šablona JSON souboru s konfigurací aplikace

Získané hodnoty konfigurace souřadnic jsou brány v metrech [m] vůči reálnému prostředí. Referenčním bodem pracovní plochy se souřadnicemi [0, 0, 0] je při pohledu z předu levý přední roh desky testbedu. Následně zadaná hodnota souřadnice k dané buňce odpovídá při pohledu z předu na buňku levému spodnímu přednímu rohu viz. Obr. 11-2. V případě markeru bod souřadnice odpovídá levému spodnímu rohu. Hodnota je zadávána ve formátu čísla s desetinnou tečkou.

V rámci konfiguračního JSON souboru je možné nastavit texturu buněk. Tuto funkci lze využít při kalibraci pozic, aby uživatel byl schopen správně nastavit pozice buněk virtuálního modelu. Nastavením *boolean* hodnoty klíče *CellsTransparent* JSON souboru, je textura zachována či nikoli.

12 POUŽITÍ SYSTÉMU

Tato kapitola vysvětluje použití aplikace ve skutečném prostředí. Jedná se o instalaci aplikace na mobilní zařízení, testování celého systému a následná možnost úpravy aplikace v případě nových požadavků a AR informační systém.

12.1 INSTALACE APLIKACE

Aplikace je vytvořena pro operační systém Android. Instalační balík *ARBarman.apk* je přiložen jako příloha a je součástí disku přiloženého k diplomové práci.

Před samotnou instalací musí být v nastavení zařízení povolena možnost instalace aplikací z neznámých zdrojů, neboť není provedena prostřednictvím obchodu Google Play. Framework ARCore, který je nedílnou součástí, požaduje nejméně Android verze 7.0 Nugat. Na stránkách ARCore je k dispozici seznam všech doposud podporovaných zařízení.

Odkaz na web:

https://developers.google.com/ar/discover/supported-devices#android_play

Pro zprostředkování dat z REST API musí být mobilní zařízení po celou dobu chodu aplikace připojené k internetu. Instalační balík aplikace musí být nejprve přesunut do uložiště telefonu, odkud lze následně spustit instalaci. Po splnění těchto podmínek lze aplikaci plně využívat.

12.2 TESTOVÁNÍ APLIKACE

V rámci závěrečného testování bylo provedeno ověření správné funkcionality celé aplikace. Testování probíhalo přímo v laboratoři na skutečném testbedu. Pouze takto bylo možné zjistit, zda aplikace vykazuje správné chování v reálném prostředí. Před samotným testováním byly měřením získány aktuální pozice buněk a následně zaneseny do konfiguračního souboru ne straně serveru.

Testovací plán byl rozdělen do několika částí, aby bylo možné ověřit dílčí subsystémy.

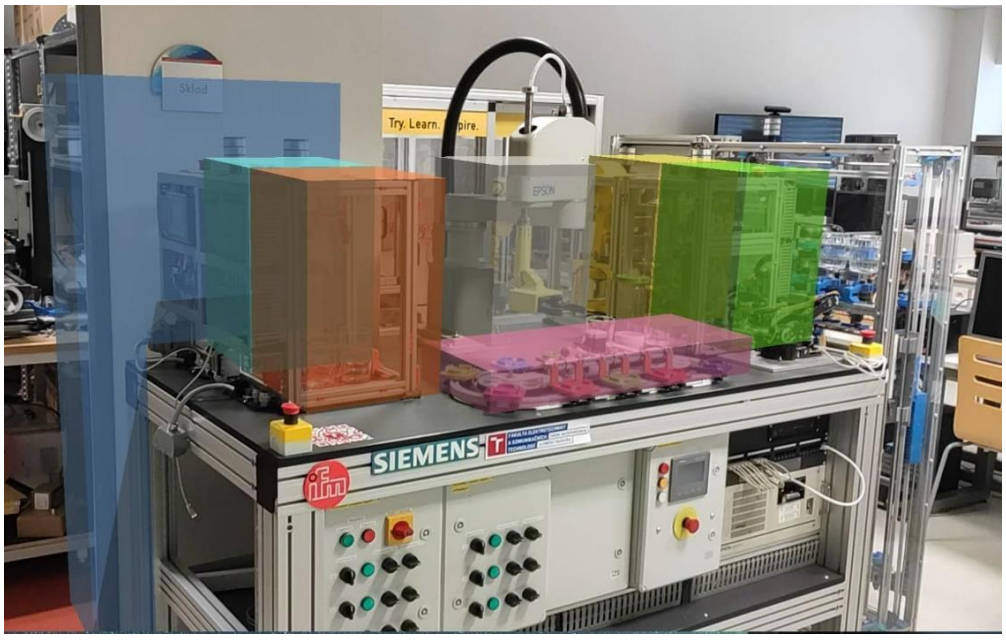
Jedná se o testování:

- uživatelského rozhraní;
- detekce markeru;
- registrace AR;
- AR informačního systému:
    - uživatelská interaktivita virtuálního modelu testbedu vůči reálnému testbedu;
    - správné vykreslení informačního systému;
- REST API:
    - získání konfigurace aplikace;
    - získání obsahu informačního systému;
    - správná interpretace obsahu v AR informačním systému.

Aplikace byla průběžně testována a laděna během vývoje, avšak mimo laboratoř a provizorním modelu testbedu. Díky závěrečným testům se skutečným testbedem bylo možné odhalit drobné nedostatky, které byly hned ošetřeny. Jednalo se například o problém s komunikací se serverem, který vracel obsah informační tabule ve špatném JSON formátu. Testování probíhalo na mobilním zařízení Xiaomi MI9 SE s operačním systémem Android 10. Následující snímky ukazují použití aplikace na reálném testbedu a vizualizují okno informačního systému a pohled na testbed s aplikovaným virtuálním modelem.

Obr. 12-1 AR informační tabule buňky Glass Storage

Obr. 12-2 Detail virtuálního modelu aplikovaného na reálný testbed



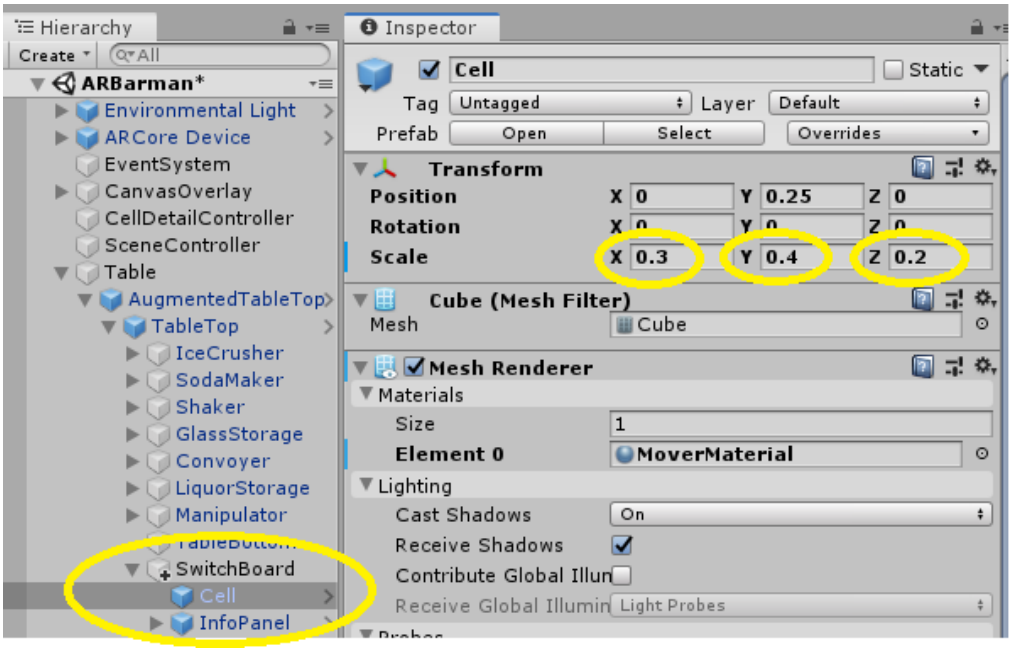
Obr. 12-3 Pohled na celý testbed s aplikovaným virtuálním modelem

12.3 PŘIDÁNÍ NOVÉ BUŇKY

V případě budoucího požadavku o rozšíření AR informačního systému o novou buňku je aplikace navržena tak, aby nebylo nutné provádět zásadní změny v aplikaci. Jako ukázka je následujícím postupem přidána nová buňka rozvaděče, který je umístěn ve spodní části testbedu.

Přidání nové buňky se skládá z těchto kroků:

1. **Vznik nové buňky** - Vytvoření nové buňky je provedeno duplikací některé ze stávajících buněk v prostředí Unity např. Soda Maker. Duplikaci lze uskutečnit v okně *Hierarchy* po kliknutí pravým tlačítkem na buňku s názvem *SodaMaker* a volbou funkce *Duplicate*. Tímto krokem je vytvořen nový herní objekt s názvem *SodaMaker (1)*, který získal veškeré vlastnosti jeho vzoru. Pro změnu rozměrů buňky, které budou odpovídat skutečné velikosti buňky v testbedu, je nutné upravit parametry v okně *Inspector* u komponenty *Transform* pro herní objekt *Cell*. Velikost skutečného rozvaděče je 30x40x20 cm, viz. Obr. 12-4.
2. **Nastavení názvu buňky –** Pojmenování buňky je zásadní pro správné načítání konfigurace pozice buňky a obsahu informační tabule ze souborů získaných z REST API. Název herního objektu buňky identifikuje element JSON souboru pro konkrétní buňku. Pro příklad byl zvolen název *SwitchBoard*. Obr. 12-4.



Obr. 12-4 Vytvořený nový herní objekt SwitchBoard a definovanými rozměry

3. **Rozšíření konfigurace z REST API** – Do konfiguračního souboru, který je načítán z REST API a určuje pozice jednotlivých buněk, je nutné přidat nastavení pro nově vzniklou buňku. Název elementu JSON souboru musí přesně odpovídat názvu herního objektu v prostředí Unity prezentující novou buňku.

```
"SwitchBoard": {
  "name": "SwitchBoard",
  "x": 0.05,
  "y": -0.45,
  "z": -0.1
}
```

Výpis 12.1 Rozšíření konfigurace z REST API

4. **Rozšíření třídy konfiguračních dat** – Do třídy *RestConfigurationData* umístěné v *Assets/Scripts* musí být přidána nová buňka se stejným názvem jako má herní objekt buňky v Unity prostředí. Tato třída slouží k vytvoření objektu konfiguračních dat z JSON souboru z REST API.

```
public class RestConfigurationData
{
    public bool CellsTransparent;
    public CellPose IceCrusher;
    public CellPose Shaker;
    public CellPose SodaMaker;
    public CellPose GlassStorage;
    public CellPose Convoyer;
    public CellPose Manipulator;
    public CellPose LiquorStorage;
    public CellPose Marker;
    public CellPose SwitchBoard; //nova bunka
}
```

Výpis 12.2 Přidání nové buňky do třídy RestConfigurationData

5. **Vytvoření REST API zdroje** – Pro nově vzniklou buňku musí být vytvořen nový REST API zdroj na straně serveru, který poskytuje obsah informačním tabulím a detailnímu náhledu buňky.

   Zdroje musí být dostupné na adresách:

   http://bb.skupra.cz/cell/cz/SwitchBoard/InfoTable/1

   http://bb.skupra.cz/cell/cz/SwitchBoard/DetailInfo/1

13 ZÁVĚR

Cílem této diplomové práce bylo vytvoření AR aplikace na operační systém Android, která integruje rozšířenou realitu do testbedu Průmysl 4.0. Myšlenkou integrace rozšířené reality do testbedu je použití principu marker-based AR a vytvoření AR informačního systému pro testbed. Všechny cíle práce se podařilo splnit, a vznikla tak plně funkční aplikace přinášející přehledný a uživatelsky interaktivní náhled v rozšířené realitě na aktuální stav jednotlivých částí testbedu. Hlavní přínos této práce je možnost aplikaci využívat při prezentování testbedu a k výukovým účelům. V případě, že v budoucnu vyvstanou nové požadavky na funkcionalitu aplikace, je možné na diplomovou práci pohodlně navázat a dále v ní pokračovat.

V úvodní kapitole 1 byla stručně popsána základní problematika, cíle a struktura práce. V kapitole 2 byl shrnut popis čtvrté průmyslové revoluce neboli Průmyslu 4.0, co ji předcházelo, a jaké má charakteristické rysy. Tuto část následuje seznámení s celou konstrukcí testbedu Průmysl 4.0.

Vysvětlení problematiky virtuality konkrétně s rozšířenou realitou a jejích přínosů v oblasti průmyslu bylo uvedeno v kapitole 3. Integrováním rozšířené reality do Barmana pomocí aplikace tak vznikl interaktivní AR informační systém testbedu.

Musely být provedeny dvě vzájemně propojené analýzy. Náplní první z nich bylo posouzení a výběr nejvhodnější AR vývojové platformy, jak je uvedeno v kapitole 4. Platforma musela být dostatečně robustní, volně dostupná a s podporou OS Android. Výsledkem analýzy byl výběr frameworku ARCore od společnosti Google popsaného v kapitole 5. Druhá analýza, která je obsahem kapitoly 6, měla za cíl výběr vývojového prostředí. V rámci této analýzy vyplynula vhodnost použití herního enginu Unity, který umožňuje multiplatformní vývoj aplikací a integruje vlastní vývojové prostředí s plnou podporou SDK ARCore. Základní pojmy a práce s Unity jsou uvedeny v kapitole 7.

Před samotnou implementací aplikace bylo nezbytné definovat všechny požadavky na vyvíjenou aplikaci, ze kterých se vycházelo v průběhu celé praktické části této práce. Východiskem aplikace je vytvoření základu projektu. Jedná se o importování a zprovoznění SDK ARCore ve vývojovém prostředí Unity a přípravu mobilního zařízení, na kterém byla aplikace během vývoje laděna. Za účelem dosažení interaktivity informačního systému v rozšířené realitě s reálným testbedem byl vytvořen zjednodušený

virtuální model testbedu v prostředí Unity, který je následně pomocí markeru aplikován na reálný testbed. Marker je umístěn na pracovní ploše testbedu a jeho úlohou je identifikace plochy testbedu ve snímané scéně a získání potřebných informací jako jsou pozice, rotace, měřítko a vzdálenost plochy vůči snímacímu zařízení pro správnou registraci rozšířené reality. Všechna výše zmíněná témata včetně vzhledu a práce s AR informačním systémem zastřešuje kapitola 8.

Uživatelské rozhraní aplikace, které je popsáno v kapitole 9, je rozděleno do dvou Unity scén. První scéna hlavního menu je zobrazena po spuštění aplikace a obsahuje nabídku pro získání informací o projektu, konfiguraci aplikace, a nakonec přechod do druhé scény aplikace s AR prostředím. Druhá scéna s AR prostředím je detailně popsána v kapitole 10.

Veškerý textový obsah aplikace je informačnímu systému předáván v definované podobě ve formátu JSON z REST API testbedu, ke kterému je nutná autentizace pomocí tokenu. REST rozhraní bylo vytvořeno v závislosti na této diplomové práci. Jak je uvedeno ve specifikaci aplikace, byl kladen velký důraz na možnost vzdáleného řízení vzhledu a obsahu informačního systému. K tomuto účelu je AR informační systém testbedu navržen tak, aby vše bylo možné řídit ze strany serveru. Řešení je popsáno v kapitole 11.

V předposlední kapitole 12 byl uveden postup instalace a zprovoznění aplikace na zařízení se systémem Android a závěrečná testování a náhled na AR aplikaci přímo na reálném testbedu. Jako poslední část kapitoly byl uveden postup přidání nové buňky v případě budoucích potřeb na rozšíření AR informačního systému o nový prvek.

SEZNAM POUŽITÉ LITERATURY

[1] Iniciativa Průmysl 4.0. In: *Ministerstvo průmyslu a obchodu* [online]. Praha, 2016 [cit. 2020-03-24]. Dostupné z: https://www.mpo.cz/assets/dokumenty/53723/64358/658713/priloha001.pdf

[2] *Technický týdeník* [online]. 2015, 2015(4) [cit. 2020-03-24]. ISSN 0040-1064. Dostupné z: https://www.technickytydenik.cz/rubriky/ekonomika-byznys/od-1-prumyslove-revoluce-ke-4_31001.html

[3] Automatizace v průmyslu I – Základy lineární techniky. In: *Haberkon* [online]. 2019 [cit. 2020-03-29]. Dostupné z: https://www.haberkorn.cz/detail/579/automatizace-v-prumyslu-i-zaklady-linearni-techniky/

[4] KACZMARCZYK, Václav, Ondřej BAŠTÁN, Zdeněk BRADÁČ a Jakub ARM. An Industry 4.0 Testbed (Self-Acting Barman): Principles and Design. *IFAC*. 2018.

[5] MILGRAM, Paul, Haruo TAKEMURA, Akira UTSUMI a Fumio KISHINO. *Augmented Reality: A class of displays on the reality-virtuality continuum*. Kyoto: ATR Communication Systems Research Laboratories, 1994.

[6] MORTON HEILIG : INVENTOR VR. In: *USC School of Cinematic Arts* [online]. [cit. 2020-03-25]. Dostupné z: http://uschefnerarchive.com/morton-heilig-inventor-vr/

[7] LeepVR history. *LeepVR* [online]. USA: LeepVR [cit. 2020-03-25]. Dostupné z: http://www.leepvr.com/history.php

[8] VAN KREVELEN, Rick. *Augmented Reality: Technologies, Applications, and Limitations: Technologies, Applications, and Limitations*. 2007. DOI: 10.13140/RG.2.1.1874.7929.

[9] Augmented Reality: Transforming Training with Immersive Experiences. In: *Readwrite* [online]. 2019 [cit. 2020-04-04]. Dostupné z: https://readwrite.com/2019/08/13/augmented-reality-transforming-training-with-immersive-experiences/

[10] How It Will Transform Logistics Industry?. In: *Https://cloudsmallbusinessservice.com/* [online]. 2018 [cit. 2020-03-29]. Dostupné z: https://cloudsmallbusinessservice.com/guest-posts/augmented-reality-apples-arkit-how-it-will-transform-logistics-industry-65473.html

[11] MUJBER, T.S, T SZECSI a M.S.J HASHMI. Virtual reality applications in manufacturing process simulation. *Journal of Materials Processing Tech* [online]. Elsevier B.V, 2004, 155-156(1-3), 1834-1838 [cit. 2020-03-25]. DOI: 10.1016/j.jmatprotec.2004.04.401. ISSN 0924-0136. Dostupné z: https://www-sciencedirect-com.ezproxy.lib.vutbr.cz/science/article/pii/S0924013604005618

[12] Fundamental concepts. *ARCore* [online]. USA: Google, 2019 [cit. 2020-03-25]. Dostupné z: https://developers.google.com/ar/discover/concepts

[13] NERURKAR, Esha, Simon LYNEN a Sheng ZHAO. *System and method for concurrent odometry and mapping*. b.r. US20170336511A1.

[14] NEUŽIL, Tomáš. *Průběžná lokalizace a mapování pomocí mobilního robotu*. Brno, 2008.. Doktorská práce. Vysoké učení technické v Brně.

[15] LEUTENEGGER, S, M CHLI a R. SIEGWART. BRISK: Binary Robust invariant scalable keypoints. In: *2011 International Conference on Computer Vision* [online]. Zurich: IEEE, 2011, s. 2548-2555 [cit. 2020-03-26]. DOI: 10.1109/ICCV.2011.6126542. ISBN 9781457711015. ISSN 15505499. Dostupné z: https://ieeexplore-ieee-org.ezproxy.lib.vutbr.cz/document/6126542

[16] Basics of AR: Anchors, Keypoints & Feature Detection. *ANDREASJAKL.COM* [online]. 2017 [cit. 2020-03-26]. Dostupné z: https://www.andreasjakl.com/basics-of-ar-anchors-keypoints-feature-detection/

[17] ROSTEN, Edward a Tom DRUMMOND. Machine Learning for High-Speed Corner Detection. *Computer Vision – ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I*. 1. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, 3951, s. 430-443. DOI: 10.1007/11744023_34. ISBN 9783540338321.

[18] TAREEN, Shaharyar a Zahra SALEEM. *A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK*. 2018. DOI: 10.1109/ICOMET.2018.8346440.

[19] CryEngine features. *Cry Engine* [online]. USA: Crytek, 2019 [cit. 2020-04-04]. Dostupné z: https://www.cryengine.com/features

[20] Unreal Engine. *UnrealEngine* [online]. USA: Epic Games, 2020 [cit. 2020-03-04]. Dostupné z: https://www.unrealengine.com/

[21] Unity Real-Time Development Platform | 3D, 2D VR & AR Visualizations. *Unity Real-Time Development Platform | 3D, 2D VR & AR Visualizations* [online]. USA: Unity Technologies, 2020 [cit. 2020-03-04]. Dostupné z: https://unity.com/

[22] Unity User Manual (2019.3). *Unity Documentation* [online]. USA: Unity Technologies, 2020 [cit. 2020-04-12]. Dostupné z: https://docs.unity3d.com/Manual/index.html

[23] TextMesh Pro. *Unity 3d docs* [online]. USA: Unity Technologies, 2019 [cit. 2020-04-23]. Dostupné z: https://docs.unity3d.com/Packages/com.unity.textmeshpro@1.3/manual/index.html#quick-start

[24] Recognize and Augment Images. *ARCore* [online]. USA: Google Developers, 2019 [cit. 2020-04-25]. Dostupné z: https://developers.google.com/ar/develop/c/augmented-images

[25] MALÝ, Martin. REST: architektura pro webové API. In: *Zdroják* [online]. Praha, 2009 [cit. 2020-05-14]. Dostupné z: https://www.zdrojak.cz/clanky/rest-architektura-pro-webove-api/

[26] SimpleJSON. *Unity Community* [online]. USA: MIT, 2017 [cit. 2020-04-23]. Dostupné z: https://wiki.unity3d.com/index.php/SimpleJSON

[27] Oculust Quest. In: *Oculus* [online]. USA: Oculus, 2020 [cit. 2020-03-25]. Dostupné z: https://www.oculus.com/

SEZNAM OBRÁZKŮ

SEZNAM ZKRATEK

IDE	-	vývojové prostředí (Integrated Development Environment)
OS	-	operační systém
SDK	-	nástroje k vývoji aplikací (Software Development kit)
OHA	-	konsorcium spravující OS Android (Open Handset Alliance)
CPS	-	kyber-fyzický systém (Cyber-Physic System)
HMD	-	náhlavní displej (Head Mouted Display)
COM	-	algoritmus souběžné odometrie a mapování (Concurent odometry and Mapping)
BRISK	-	Binary Robust Invariant Scalable Keypoints
SLAM	-	Simultaneous localization and mapping
FAST	-	Features From Accelerated Segment Test
GNU	-	General Public Licence

SEZNAM TABULEK

SEZNAM PŘÍLOH

Příloha 1: Ukázka JSON souboru s obsahem informační tabule pro buňku (součástí textových příloh i přiložené CD)

Příloha 2: Ukázka JSON souboru s obsahem informační tabule pro buňku (součástí textových příloh i přiložené CD)

Příloha 3: Elektronická verze diplomové práce (přiložené CD)

Příloha 4: Instalační balík aplikace *ARBarman.apk* (přiložené CD)

Příloha 5: Balík *ARBarman_xpolac32.unitypackage* - kompletní projekt (přiložené CD)

PŘÍLOHY

Příloha 1

Ukázka JSON souboru s obsahem informační tabule pro buňku

```
{
   "language":"CZ",
   "text1":{
      "text":"<align=center>Ventil 1\n<size=80><sprite=25></size>\n\nVen-
til 2\n<size=80><sprite=26></size>\n\nVen-
til 3\n<size=80><sprite=26></size>\n\nServo 1\n<size=80><sprite=27></size></align>",
      "width":30,
      "high":80
   },
   "text2":{
      "text":"<align=center><size=60>STAV ZÁSOBNÍKŮ</size>\n\nHladina zásob-
níku1\n<size=200><sprite=2></size>\n\nHladina zásob-
níku2\n<size=200><sprite=17></size>\n\nHladina zásob-
níku3\n<size=200><sprite=13></size>",
      "width":70,
      "high":80
   },
   "text3":{
      "text":"<align=center><size=60><color=yellow>Chy-
bové stavy</size></color></align>\n<size=70><sprite=23> </size> Všechny kompo-
nenty pracují správně a zařízení je připraveno\n<size=70><sprite=22> </size> Do-
chází náplň <b>zásobníku 1</b>\n<size=70><sprite=24> </size> Dochází náplň <b>zásob-
níku 1</b>\n<size=70><sprite=21> </size> Bude potřeba sevisní prohlídka v příš-
tím spuštění\n",
      "width":100,
      "high":20
   },
   "buttonUpPage":true,
   "buttonDownPage":false
}
```

Příloha 2

Ukázka JSON souboru s konfigurací virtuálního modelu testbedu

```
{
  "CellsTransparent": true,
  "IceCrusher": {
    "name": "IceCrusher",
    "x": 0.15,
    "y": 0,
    "z": 0.65
  },
  "Shaker": {
    "name": "Shaker",
    "x": 1.48,
    "y": 0,
    "z": 0.65
  },
  "SodaMaker": {
    "name": "SodaMaker",
    "x": 1.48,
    "y": 0,
    "z": 0.25
  },
  "GlassStorage": {
    "name": "GlassStorage",
    "x": 0.15,
    "y": 0,
    "z": 0.25
  },
  "Convoyer": {
    "name": "Convoyer",
    "x": 0.65,
    "y": 0,
    "z": 0.2
  },
  "Manipulator": {
    "name": "Manipulator",
    "x": 0.835,
    "y": 0,
    "z": 0.6
  },
  "LiquorStorage": {
    "name": "LiquorStorage",
    "x": -0.15,
    "y": 0,
    "z": 1.1
  },
  "Marker": {
    "name": "Marker",
    "x": 0,
    "y": 0,
    "z": 0
  }
}
```